

AACL-IJCNLP 2020

**The 1st Conference of the Asia-Pacific Chapter of the  
Association for Computational Linguistics and  
the 10th International Joint Conference on  
Natural Language Processing**

**Proceedings of the Conference**

December 4 - 7, 2020

**Diamond Level Sponsor**



**Diversity & Inclusion: Champion Sponsor**

**Bloomberg**

**Engineering**

**Diversity & Inclusion: In-Kind Sponsor**



**Remote Presentation Sponsor**



©2020 The Association for Computational Linguistics

Order copies of this and other ACL proceedings from:

Association for Computational Linguistics (ACL)  
209 N. Eighth Street  
Stroudsburg, PA 18360  
USA  
Tel: +1-570-476-8006  
Fax: +1-570-476-0860  
[acl@aclweb.org](mailto:acl@aclweb.org)

ISBN 978-1-952148-91-0

## Message from the General Chair

Welcome to the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing (AACL-IJCNL-2020) virtually hosted by Soochow University, China. I am honored to be the General Chair of the first conference of AACL and am excited to welcome you to the event.

Research in NLP in the Asia-Pacific region is rapidly growing. This is evident from the increasing number of paper submissions and participants in recent NLP conferences worldwide. Many innovative NLP theory and applications are presented in these events. To better share NLP innovations in the region, the Association for Computational Linguistics (ACL) established the Asia-Pacific Chapter of (AACL) in 2018. AACL provides a regional focus for ACL members in Asia-Pacific region to promote and to facilitate cooperation and information exchange among related scientific and professional societies and individuals in the region. This is the first bi-annual meeting of the Chapter and is jointly organized with the 10th International Joint Conference on Natural Language Processing (IJCNLP), which is the bi-annual conference of the Asia Federation of Natural Language Processing (AFNL), a well-established association sharing the same missions as AACL. There is a saying “1+1 > 2” in China and this joint conference truly reflects that. AACL-IJCNLP-2020 is just a beginning, AACL will initiate more collaborations with AFNLP and other regional associations in the future so as to let the world appreciate more the work of the NLP communities in Asia-Pacific.

Soochow is a scenic and historic Chinese city. The city's canals, stone bridges, pagodas and meticulously designed gardens have contributed to its status as one of the top tourist attractions in China. Noticeably, the Classical Gardens of Soochow were added to the list of the UNESCO World Heritage Sites in 1997 and 2000. Soochow is often dubbed the “Venice of the East” or "Venice of China". These attractions would have added another flavor to the AACL-IJCNLP-2020 blending it a cultural and technical event. Several cultural tours and social events were originally planned. They would have been eye-opening to the participants. However, a face-to-face physical conference gave way to the COVID-19 pandemic. After a long period of observation since the beginning of the year, due to the growing severity of the pandemic worldwide, the local organizing committee recommended to organize AACL-IJCNL-2020 virtually in September. This was a difficult decision; but for safety reason, the team and I believed this was the right choice. Organization of a virtual conference was unconventional. It created a new set of challenging problems unfamiliar to the team. Fortunately, ACL2020 took place before AACL-IJCNLP-2020 and we learned much from their experience.

The program co-chairs Kevin Knight and Hua Wu did an excellent job in putting together a very interesting program with 92 papers from submissions worldwide. The program also includes two star keynote speakers, Percy Liang, Stanford University, and Song-Chun Zhu, Peking University, Tsinghua University, and UCLA, to share with us their insights in “semantic parsing” and “explainable AI” respectively. These are hot topics both in pure and applied research. Selecting the papers from nearly 400 submissions was itself a difficult task but it was compounded by the complication in the design of the virtual program. Kevin and Hua had to take into account of the different time zones of the authors and audience and was rather tricky. Nevertheless, after many iterations, they came up with the current exciting program.

The conference is accompanied by 7 workshops and 6 tutorials. We participated in the joint selection processes with other ACL related conferences. The workshop co-chairs, Wei Gao and Lu Wang as well as the tutorial co-chairs Timothy Baldwin and Fei Xia did a wonderful job to select these very educational and trendy topics. In addition, despite the busy program schedule, with the dedication of the student workshop co-chairs, Lun-Wei Ku and Vincent Ng, we also put up a student research workshop; and with the dedication of the demo co-chairs, Douwe Kiela and Derek Wong, we accepted 7 system projects

out of 15 for demonstration. As a part of ACL's Diversity and Inclusion (D&I) initiative, we introduced the Widening NLP (WiNLP) session in the first day of the conference. Xiangyu Duan and Tirthankar Ghosal, the D&I co-chairs, were very creative in organizing the session and other virtual D&I gathering events.

The publication co-chairs Steve DeNeefe and Satoshi Sekine worked diligently and carefully to collect the accepted papers for compilation of the electronic proceedings. Following the ACL guidelines, the papers are also included in the ACL Anthology. Since AACL-IJCNLP-2020 is a virtual conference, we decided not to compile a physical handbook. As such, information about the conference is made available online.

The local organization committee was chaired by Min Zhang. As usual, LOC is the committee which does all the ‘dirty’ jobs, ie hands-on work. Under Min’s leadership, LOC did a fantastic job in looking after all the details in local arrangement. Organizing a virtual conference was new to Min and the LOC. They put extra time and effort to ensure every single organization details were properly arranged. I also appreciated the frustration of the LOC at the beginning. They booked all the venues while the spreading of the COVID19 pandemic showed no sign of slowing down. We waited until September before we decided not to go for the face-to-face option, which left us with rather little time to prepare for the virtual conference. I am glad that we finally make it.

Working closely with the LOC, the remote presentation co-chairs, Nanyun Peng, Zhongqing Wang and Muyun Yang liaised with the program committee, demonstration committee, workshop committee, and tutorial committee to ensure presentations would be done smoothly. The webmaster, Junhui Li, was also part of the LOC, to ensure information related to the conference was timely distributed. Mirella Lapata, Haizhou Li and Qun Liu, the publicity co-chairs, made use of this conference information for international publicity through different channels, eg ACL, SIGIR, AAAI etc communities, and media. Eg. Instagram, Twitter, Facebook, Email, Wechat, etc.

Last but not the least, I would like to thank Priscilla Rasmussen, business manager of ACL HQ, for her invaluable advice throughout the AACL-IJCNLP-2020 project. She is definitely the most knowledgeable person in ACL event matters. And I am grateful to Haifeng Wang, President of AACL, and Chengqing Zong, President of AFNLP, for their trust. Undoubtedly, without their unfailing support, AACL-IJCNLP-2020 would not be possible. Also, thanks are due to Baidu and Huawei, the Diamond level sponsor, as well as Bloomberg and Grammarly for their generosity in supporting our D&I drive.

Enjoy AACL-IJCNLP-2020!

Kam-Fai Wong  
General Chair  
Hong Kong.

## **Message from the Program Chairs**

Greetings, and welcome to AACL-IJCNLP 2020, the First Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (ACL) and the Tenth International Joint Conference on Natural Language Processing (IJCNLP)!

The Asia-Pacific Chapter of ACL is the newest chapter of ACL, and its first conference was held in 2020. The AACL-IJCNLP 2020 conference received 392 submissions to the main conference. We accepted 106 papers (73 long, 33 short), for an overall 28.3% acceptance rate. Submissions came from all over the world. Of the 106 accepted papers, 53 were from the Asia-Pacific region (28 from China, 10 from Japan, 7 from India, 3 from Taiwan, 2 from Australia, 2 from South Korea, and 1 from Indonesia), 31 from the Americas (27 from the USA, 3 from Canada, 1 from Brazil), and 22 from Europe and the Middle East (9 from the UK, 5 from Germany, 2 from France, 2 from Israel, 1 from the Netherlands, 1 from Italy, 1 from Denmark, and 1 from Belgium). 14 accepted papers were withdrawn by the authors, due to simultaneous submission to other conferences, resulting in a total of 92 papers presented at the main conference of AACL-IJCNLP 2020.

Like most conferences in 2020, ACL was a virtual one. Conference organizers opted for a real-time meeting, with papers grouped into topical sessions as done previously for in-person conferences. Authors further pre-recorded talks for convenient, off-schedule viewing.

We had two keynote addresses this year, one by Percy Liang of Stanford University, and one by Song-Chun Zhu of Peking University, Tsinghua University, and UCLA. Many thanks to Professors Liang and Zhu for exciting the AACL-IJCNLP participants with their sweeping talks!

For the main conference, 46 Area Chairs oversaw 392 submissions to 18 tracks. We would like to thank the Area Chairs and reviewers for their hard work. In addition, thanks to Rich Gerber of SoftConf for his always-timely help and Natalie Schluter for sharing the ACL 2020 paper-review matching tool with us. We would also like to thank the General, Local, and Publication Chairs for guidance and assistance in creating the main conference program. Finally, many thanks to the authors who carried out the research and submitted their work to AACL-IJCNLP 2020. We hope that the new AACL will continue to be a vibrant place to exchange research and ideas!

Kevin Knight, DiDi Labs  
Hua Wu, Baidu  
ACM-IJCNLP 2020 Program Committee Co-Chairs

## **Organizing Committee**

### **General Chair:**

Kam-Fai Wong, The Chinese University of Hong Kong, Hong Kong SAR, China

### **Program Committee Co-Chairs:**

Kevin Knight, Didi Chuxing, USA  
Hua Wu, Baidu, China

### **Organisation Chair:**

Min Zhang, Soochow University, China

### **Workshop Co-Chairs:**

Wei Gao, Singapore Management University, Singapore  
Lu Wang, Northeastern University, USA

### **Student Workshop Co-Chair:**

Lun-Wei Ku, IIS, Academic Sinica, Taiwan  
Vincent Ng, University of Texas at Dallas, USA

### **Tutorial Co-Chairs:**

Fei Xia, University of Washington, USA  
Timothy Baldwin, University of Melbourne, Australia

### **Demo Co-Chairs:**

Derek Wong, University of Macau, Macau SAR, China  
Douwe Kiela, Facebook, USA

### **Publication Co-Chairs:**

Satoshi Sekine, RIKEN, AIP, Japan  
Steve DeNeefe, SDL, USA

### **Publicity Co-Chairs:**

Qun Liu, Huawei Noah's Ark Lab, Hong Kong SAR, China  
Haizhou Li, National University of Singapore, Singapore

Mirella Lapata, University of Edinburgh, UK

**Sponsorship Co-ordinators:**

Ming Zhou, Microsoft Research Asia, China  
Yusuke Miyao, University of Tokyo, Japan

**Diversity & Inclusion Co-Chairs:**

Tirthankar Ghosal, Indian Institute of Technology Patna, India  
Xiangyu Duan, Soochow University, China

**Webmaster:**

Junhui Li, Soochow University, China

**AACL CCC Representative:**

Yang Liu, Tsinghua University, China

**Remote Presentation Co-Chairs:**

Zhongqing Wang, Soochow University, China  
Muyun Yang, Harbin Institute of Technology, China  
Nanyun Peng, Information Sciences Institute, USA

## **Program Committee**

### **Program Committee Co-Chairs:**

Kevin Knight, Didi Chuxing  
Hua Wu, Baidu, China

### **Area Chairs:**

#### **Dialogue and Interactive Systems**

Michel Galley, Minglie Huang

#### **Discourse and Pragmatics**

Sujian Li, Vincent Ng, Michael Strube

#### **Information Retrieval and Document Analysis**

Hitoshi Isahara, Yiqun Liu

#### **Information Extraction and Text Mining**

Dan Bikel, Hsin-Hsi Chen, Heng Ji

#### **Knowledge Graphs**

Sujith Ravi, Jie Tang

#### **Linguistic Theories, Cognitive Modeling and Psycholinguistics**

Mark Hopkins, William Schuler

#### **Machine Learning for NLP**

Kevin Duh, Wei Lu, Jun Suzuki

#### **Machine Translation and Multilinguality**

Marta R. Costa-jussà, Zhongjun He, Jonathan May, Taro Watanabe

#### **NLP Applications**

Jiwei Li, Lei Li

#### **Phonology, Morphology and Word Segmentation**

Daisuke Kawahara, Xipeng Qiu

#### **Question Answering**

Danqi Chen, Kang Liu, Scott Yih

#### **Resources and Evaluation**

Laura Rimell, Nianwen Xue

#### **Semantics**

Yaser Al-Onaizan, Kai-Wei Chang, He He, Michael Roth

#### **Sentiment Analysis and Argument Mining**

Jinho Choi, Yue Zhang

#### **Social Media**

Eiji Aramaki, Zhiyuan Liu, Alice Oh

**Summarization and Generation**

Mark Dras, Xiaojun Wan, Lu Wang

**Speech, Vision, Robotics, Multimodal Grounding**

Taylor Berg-Kirkpatrick, Yonatan Bisk

**Tagging, Chunking, Syntax and Parsing**

Wanxiang Che, Shay Cohen

**Primary Reviewers:**

Mourad Abbas, Asad Abdi, Mostafa Abdou, Omri Abend, Abhishek Abhishek, Sallam AbuAlhaija, Abdalghani Abujabal, Lasha Abzianidze, Yvonne Adesam, Vaibhav Adlakha, Oshin Agarwal, Rodrigo Agerri, Pius Aggarwal, Željko Agić, Priyanka Agrawal, Roee Aharoni, Benyamin Ahmadnia, Aman Ahuja, Chaitanya Ahuja, Akiko Aizawa, Mohammad Akbari, Alan Akbik, Farhad Akhbardeh, Md. Shad Akhtar, Syed Sarfaraz Akhtar, Nader Akoury, Khalid Al Khatib, Firoj Alam, Chris Alberti, Nikolaos Aletras, Fahad AlGhamdi, Bashar Alhafni, Haifa Alharthi, Hamed Alhoori, Ahmed Ali, Hend Al-Khalifa, Hussein Al-Olimat, Yaser Al-Onaizan, Miguel A. Alonso, Nora Al-Twairesh, Fernando Alva-Manchego, Reinald Kim Amplayo, Guozhen An, Jisun An, Raviteja Anantha, Antonios Anastasopoulos, Anietie Andy, Mohammed Ansari, João António Rodrigues, Emilia Apostolova, Jun Araki, Rahul Aralikatte, Eiji ARAMAKI, Yuki Arase, Mihael Arcan, John Arevalo, Fawaz Arfaj, Arturo Argueta, Naveen Arivazhagan, Piyush Arora, Ekaterina Artemova, Masayuki Asahara, Ehsaneddin Asgari, Nabiha Asghar, Elliott Ash, Lukasz Augustyniak, Eleftherios Avramidis, Amar Prakash Azad

Rohit Babbar, Nguyen Bach, Ebrahim Bagheri, Ke Bai, Xuefeng Bai, JinYeong Bak, Amir Bakarov, Mithun Balakrishna, Timothy Baldwin, Miguel Ballesteros, Ramy Baly, Sivaji Bandyopadhyay, Sameer Bansal, Guangsheng Bao, Ankur Bapna, Mohamad Hardyman Barawi, Jeremy Barnes, Solon Barocas, Pierpaolo Basile, Valerio Basile, Mohaddeseh Bastan, Tanmay Basu, Timo Baumann, Hanna Bechara, Lee Becker, Barend Beekhuizen, Gašper Beguš, Núria Bel, Yonatan Belinkov, Eric Bell, Emily M. Bender, Fernando Benites, Gábor Berend, Taylor Berg-Kirkpatrick, Sabine Bergler, Toms Bergmanis, Gabriel Bernier-Colborne, Thales Bertaglia, Dario Bertero, Robert Berwick, Laurent Besacier, Rahul Bhagat, Archna Bhatia, Parminder Bhatia, Arnab Bhattacharya, Sudha Bhingardive, Bin Bi, Wei Bi, Dan Bikel, Yi Bin, Arne Binder, Yonatan Bisk, Debmalya Biswas, Henrik Björklund, Johanna Björklund, Philippe Blache, Damián Blasi, Nate Blaylock, Su Lin Blodgett, Michael Bloodgood, Victoria Bobicev, Sravan Bodapati, Reihane Boghrati, Danushka Bollegala, Daniele Bonadiman, Antonio Bonafonte, Claudia Borg, Aurélien Bossard, Antoine Bosselut, Nadjet Bouayad-Agha, Florian Boudin, Pierrette Bouillon, Zied Bouraoui, Siddhartha Brahma, Ana Brassard, Chloé Braud, Chris Brockett, Thomas Brovelli (Meyer), Caroline Brun, Amar Budhiraja, Alberto Bugarín Díz, Trung Bui, Wray Buntine, Benjamin Börschinger

S C, José G. C. de Souza, Elena Cabrio, Deng Cai, Yi Cai, YUNFENG CAI, Andrew Caines, Ruket Cakici, Agostina Calabrese, Mary Elaine Califf, John Calvo Martinez, Erik Cambria, William Campbell, Ricardo Campos, Ed Cannon, Kris Cao, Yixin Cao, Yuan Cao, Ziqiang Cao, Cornelia Caragea, Marine Carpuat, Giovanni Cassani, Vittorio Castelli, Giuseppe Castellucci, Thiago Castro Ferreira, Chundra Cathcart, Paulo Cavalin, Fabio Celli, Daniel Cer, Alessandra Cervone, Özlem Çetinoğlu, Mauro Cettolo, Tuhin Chakrabarty, Tanmoy Chakraborty, Yllias Chali, Jon Chamberlain, Hou Pong Chan, Yee Seng Chan, Ashis Chanda, Sarath Chandar, Senthil Chandramohan, Khyathi Raghavi Chandu, Angel Chang, Franklin Chang,

Kai-Wei Chang, Yung-Chun Chang, Soravit Changpinyo, Ravikiran Chanumolu, Rochana Chaturvedi, Geeticka Chauhan, Wanxiang Che, Baoyang Chen, Bin Chen, Bo Chen, Boxing Chen, Chung-Chi Chen, Danqi Chen, Feiyang Chen, Francine Chen, Fuxiang Chen, Guanyi Chen, Howard Chen, Hsin-Hsi Chen, Huajun Chen, Huimin Chen, Huiyuan Chen, John Chen, JUNYA CHEN, Junying Chen, Kehai Chen, Kuan-Yu Chen, Long Chen, Lu Chen, Luoxin Chen, MeiHua Chen, MuHao Chen, Penghe Chen, Ping Chen, Qian Chen, Qiang Chen, Sihao Chen, Tongfei Chen, Wenhua Chen, Wenliang Chen, Xiaoli Chen, Xinchi Chen, Xinyun Chen, Xiuying Chen, Yanping Chen, Yubo Chen, Yue Chen, Yufeng Chen, Yun Chen, Yun-Nung Chen, Zhumin CHEN, Fei Cheng, Hao Cheng, Jianpeng Cheng, Pu-Jen Cheng, Weiwei Cheng, Vijil Chenthamarakshan, Colin Cherry, Emmanuele Chersoni, Jackie Chi Kit Cheung, Lianhua Chi, Jen-Tzung Chien, Hai Leong Chieu, Dhivya Chinnappa, Luis Chiruzzo, Eunah Cho, Hyundong Cho, Eleanor Chodroff, Jinho D. Choi, Kostadin Cholakov, Shamil Chollampatt, Leshem Choshen, Monojit Choudhury, Shammur Absar Chowdhury, Christos Christodoulopoulos, Chenhui Chu, Christopher Chu, Hsiao-Min Chuang, Jin-Woo Chung, Kenneth Church, Abu Nowshed Chy, Philipp Cimiano, Alina Maria Ciobanu, Elizabeth Clark, Oana Cocarascu, Arman Cohan, Shay B. Cohen, William Cohen, Çağrı Çöltekin, Costanza Conforti, John Conroy, Mathieu Constant, Danish Contractor, Paul Cook, Bonaventura Coppola, Francesco Corcoglioniti, Marta R. Costa-jussà, Josep Crego, Inés Crespo, Danilo Croce, Heriberto Cuayahuitl, Lei Cui, Shaobo Cui, Yiming Cui, Rossana Cunha, Anna Currey, Tonya Custis

Raj Dabre, Na Dai, Xinyu Dai, Zeyu Dai, Zhuyun Dai, Daniel Dakota, bharath dandala, Ankit Dangi, Falavigna Daniele, Amitava Das, Avisha Das, Dipanjan Das, Dipankar Das, Pradipto Das, Mithun Das Gupta, Tirthankar Dasgupta, Pradeep Dasigi, Vidas Daudaravicius, Tobias Daudert, Gaël de Chalendar, Éric de la Clergerie, Miryam de Lhoneux, Renato De Mori, Valeria de Paiva, Thierry Declerck, Mathieu Dehouck, Luciano Del Corro, Rodolfo Delmonte, Louise Deléger, Vera Demberg, Seniz Demir, Carrie Demmans Epp, Steve DeNeefe, Lingjia Deng, Shumin Deng, Yuntian Deng, Zhi-Hong Deng, Leon Derczynski, Shrey Desai, Nina Dethlefs, Chris Develder, Barry Devereux, Kunal Dey, Bhuwan Dhingra, Maria Pia di Buono, Luigi Di Caro, Mona Diab, Gaël Dias, Dennis Diefenbach, Jana Diesner, Chenchen Ding, Haibo Ding, Kaize Ding, Shuoyang Ding, Weicong Ding, Xiao Ding, Kalpit Dixit, Nemanja Djuric, Quynh Do, Tobias Domhan, Miguel Domingo, Li Dong, Shichao Dong, Tiansi Dong, Yuxiao Dong, Dejing Dou, Zhicheng Dou, A. Seza Doğruöz, Mark Dras, Rotem Dror, Aleksandr Drozd, Jinhua Du, Lan Du, Xinya Du, Junwen Duan, Xiangyu Duan, Pablo Duboue, Christian Dugast, Kevin Duh, Anca Dumitrache, Ewan Dunbar, Jonathan Dunn, Ondřej Dušek

Hiroshi Echizen'ya, Sauleh Eetemadi, Thomas Effland, Steffen Eger, Markus Egg, Koji Eguchi, Vladimir Eidelman, Jacob Eisenstein, Asif Ekbal, Layla El Asri, Ahmed El Kholy, Ismail El Maarouf, Randa Elanwar, Wassim El-Hajj, Micha Elsner, Messina Enza, Alexander Erdmann, Akiko Eriguchi, Liana Ermakova, Arash Eshghi, Ramy Eskander, Luis Espinosa Anke, Miquel Esplà-Gomis, Diego Esteves, Andrea Esuli, Keelan Evanini

Marzieh Fadaee, Mauro Falcone, Ingrid Falk, Tobias Falke, Kai Fan, Licheng Fang, Zheng Fang, M. Amin Farajian, Marcello Federico, Hao Fei, Fuli Feng, Shi Feng, Shi Feng, Xiaocheng Feng, Yansong Feng, Paulo Fernandes, Daniel Fernández-González, Emilio Ferrara, Olivier Ferret, Elisabetta Fersini, Oluwaseyi Feyisetan, Alejandro Figueroa, Mark Finlayson, Orhan Firat, Mauajama Firdaus, Mark Fishel, Margaret Fleck, José A. R. Fonollosa, Tommaso Fornaciari, Karén Fort, Abdellah Fourtassi, Thomas François, Diego Frassinelli, Dayne Freitag, Markus Freitag, André Freitas, Larissa Freitas, Daniel Fried, Annemarie Friedrich, Guohong Fu, Liye Fu, Xuandi FU, Zhenxin Fu, Zuohui Fu, Hagen Fuerstenau, Akinori Fu-

jino, Atsushi Fujita, Nancy Fulda, Michael Färber

Byron Galbraith, Michel Galley, Björn Gambäck, Michael Gamon, Leilei Gan, Zhe Gan, Debasis Ganguly, Niloy Ganguly, Cuiyun Gao, Peng Gao, Shen Gao, Wei Gao, Yang Gao, Utpal Garain, Jesús Miguel García-Gorrostieta, Andrew Gargett, Ekaterina Garmash, Guillermo Garrido, Dragan Gasevic, Lorenzo Gatti, Susan Gauch, Dipesh Gautam, Lieke Gelderloos, Debela Gemechu, Daniela Gerz, Reza Ghaeini, Tirthankar Ghosal, Kripabandhu Ghosh, Samujjwal Ghosh, Sucheta Ghosh, George Giannakopoulos, David Gillespie, Kevin Gimpel, Rahul Goel, Lorraine Goeuriot, Lukasz Golab, Behzad Golshan, Jose Manuel Gomez-Perez, Hongyu Gong, Jesús González-Rubio, Colin Gordon, Genevieve Gorrell, Isao Goto, Cyril Goutte, Anuj Goyal, Kartik Goyal, Pawan Goyal, Natalia Grabar, Yulia Grishina, Andreas Grivas, Stig-Arne Grönroos, Jiatao Gu, Jiuxiang Gu, Yanhui Gu, Yi Guan, Imane Guellil, Tao Gui, Kalpa Gunaratna, Tunga Gungor, Ruocheng Guo, Xiaoxiao Guo, Yuhang Guo, Abhinav Gupta, Pankaj Gupta, Shashank Gupta, Vivek Gupta, Iryna Gurevych, Joakim Gustafson, Yoan Gutiérrez, Jeremy Gwinnup, Carlos Gómez-Rodríguez

Jung-Woo Ha, Thanh-Le Ha, Nizar Habash, Ivan Habernal, Hatem Haddad, Barry Had-dow, Asmelash Teka Hadgu, Gholamreza Haffari, Masato Hagiwara, Zhen Hai, Thomas Hain, Felix Hamborg, Michael Hammond, LIFENG HAN, Tian Han, Ting Han, Xianpei Han, Xu Han, Xudong Han, Abram Handler, Jie Hao, Junheng Hao, Tianyong Hao, Yuexing Hao, Md Enamul Haque, Rejwanul Haque, Giannis Haralabopoulos, Christian Hardmeier, Mareike Hartmann, Sadid A. Hasan, Maram Hasanain, Mohammed Hasanuzzaman, Chikara Hashimoto, Claudia Hauff, Annette Hautli-Janisz, Catherine Havasi, Hiroaki Hayashi, Yoshi-hiko Hayashi, Shirley Anugrah Hayati, Ben He, Bin He, Daqing He, Hao He, He He, Jun He, Shexia He, Shilin He, Wei He, Yifan He, Zhengqiu He, Zhongjun He, Kenneth Heafield, Michael Heck, Benjamin Heinzerling, Simon Hengchen, Nico Herbig, Delia Irazú Hernández Farías, Daniel Hershcovich, Jonathan Herzog, Jack Hessel, Christopher Hidey, Felix Hieber, Derrick Higgins, Tsutomu Hirao, Sorami Hisamoto, Barbora Hladka, Tin Kam Ho, Cong Duy Vu Hoang, Hieu Hoang, Eben Holderness, Nora Hollenstein, Laura Hollink, Chester Holtz, Ari Holtzman, ZAN HONGYING, Mark Hopkins, Enamul Hoque, Ales Horak, Chiori Hori, Feng Hou, Lei Hou, Yufang Hou, Shu-Kai HSIEH, Chao-Chun Hsu, Changjian Hu, han hu, Pengwei Hu, Po Hu, Qinmin Vivian Hu, Renfen Hu, Wei Hu, Yuheng Hu, Hang Hua, Wenyue Hua, Xinyu Hua, Chung-Chi Huang, Dandan Huang, Hen-Hsen Huang, Jiaji Huang, Jiangping Huang, Jing Huang, Lifu Huang, Minlie Huang, Ruihong Huang, Shujian Huang, Songfang Huang, Xiaolei Huang, Xuanjing Huang, Yi-Ting Huang, Patrick Huber, Matthias Huck, Kai Hui, Muhammad Humayoun, Samar Husain, Rebecca Hwa, Sung Ju Hwang, Ali Hürriyetoğlu

Ignacio Iacobacci, Ebuka Ibeke, Adrian Iftene, Ryu Iida, Filip Ilievski, Dmitry Ilvovsky, Kenji Imamura, Aizhan Imankulova, Chase Inguva, Koji Inoue, Naoya Inoue, Kentaro Inui, Takashi Inui, Radu Tudor Ionescu, Mikel Iruskieta, Hitoshi Isahara, Hayate Iso, Alexei V. Ivanov, Lubomir Ivanov, Julia Ive

Guillaume Jacquet, Kokil Jaidka, Minni Jain, Prachi Jain, Renu Jain, Mona Jalal, Shoaib Jameel, Abhik Jana, Hyeju Jang, Slava Jankin, Hwisang Jeon, Elisabetta Jezek, Girish Jha, Harsh Jhamtani, Donghong Ji, Feng Ji, Heng Ji, Xiaowen Ji, Yangfeng Ji, Zongcheng Ji, Robin Jia, Weijia Jia, Yuxiang Jia, Ping Jian, Di Jiang, Jyun-Yu Jiang, Meng Jiang, Wenbin Jiang, Xin Jiang, Yong Jiang, Zhuolin Jiang, Zhuoren Jiang, Zhuoxuan Jiang, Pengfei Jiao, Zhanming Jie, Di Jin, Hongxia Jin, Lifeng Jin, Peng Jin, Baoyu Jing, Hwiyeol Jo, alexander johansen, Melvin Johnson, Pamela Jordan, Aditya Joshi, Dhanya Jothimani, Kyomin Jung

Vimal Kumar K, Besim Kabashi, Jad Kabbara, Kyo Kageura, Indika Kahanda, Emmanuel Kahembwe, Tomoyuki Kajiwara, Surya Kallumadi, Hirotaka Kameko, Min-Yen Kan, Katharina Kann, Diptesh Kanojia, Dain Kaplan, Sudipta Kar, Pinar Karagoz, Mladen Karan, Sarvnaz Karimi, Payam Karisani, Börje Karlsson, Shubhra (Santu) Karmaker, Sanjeev Kumar Karn, Omid Kashefi, Makoto P. Kato, David Kauchak, Daisuke Kawahara, Hideto Kazawa, Ruth Kempson, Casey Kennington, Katia Lida Kermanidis, Kimmo Kettunen, Madian Khabsa, Salam Khalifa, Mohammed Khalilia, Alizishaan Khatri, Chandra Khatri, Ashiqur KhudaBukhsh, Douwe Kiela, Halil Kilicoglu, Doo Soon Kim, Gunhee Kim, Joo-Kyung Kim, Jung-Jae Kim, Seokhwan Kim, Sun Kim, Sunghwan Mac Kim, Yunsu Kim, David King, Tracy Holloway King, Christo Kirov, Chunyu Kit, Norihide Kitaoka, Shun Kiyono, Bennett Kleinberg, Roman Klinger, Julien Kloetzer, Kevin Knight, Rebecca Knowles, Akio Kobayashi, Thomas Kober, Philipp Koehn, Mamoru Komachi, Kazunori Komatani, Kanako Komiya, Rik Koncel-Kedziorski, Xiang Kong, Myoung-Wan Koo, Valia Kordon, Yannis Korkontzelos, Punit Singh Koura, Venelin Kovatchev, Zornitsa Kozareva, Pavel Kral, Sebastian Krause, Ralf Krestel, Julia Kreutzer, Rajasekar Krishnamurthy, Nikhil Krishnaswamy, Udo Kruschwitz, Seth Kulick, malhar kulkarni, Gaurav Kumar, Shankar Kumar, Sumeet Kumar, vishwajeet kumar, Jonathan K. Kummerfeld, Florian Kunneman, C.-C. Jay Kuo, Murathan Kurfali, Vinod Kumar Kurmi, Mucahid Kutlu, Haewoon Kwak, Arne Köhn

Gorka Labaka, Matthieu Labeau, Ophélie Lacroix, Mathieu Lafourcade, Chiraag Lala, Divesh Lala, John P. Lalor, Albert Y.S. Lam, Wai Lam, Guy Lapalme, Ekaterina Lapshinova-Koltunski, Stefan Larson, Jey Han Lau, Alberto Lavelli, Carolin Lawrence, Dawn Lawrie, Phong Le, Joseph Le Roux, Gianluca Lebani, Chong Min Lee, I-Ta Lee, John Lee, Joseph Lee, Lung-Hao Lee, Minwoo Lee, Moontae Lee, Roy Ka-Wei Lee, Young-Suk Lee, Gurpreet Lehal, Wenqiang Lei, Zeyang Lei, Jochen L. Leidner, Gaël Lejeune, Alessandro Lenci, Yves Lepage, Omer Levy, Martha Lewis, Baoli Li, Binyang Li, Bo Li, Bo Li, Changliang Li, Chen Li, Chenliang Li, Dingcheng Li, Fangtao Li, Haibo Li, Haizhou Li, Hang Li, Haoran Li, Hongzheng Li, Jiaqi Li, Jinchao Li, Jing Li, Jinpeng Li, Jiwei Li, Juan Li, Juntao Li, Junyi Li, Junyi Jessy Li, Lei Li, Liangyou Li, Lishuang Li, Maolin Li, Peifeng Li, Peng Li, Peng-Hsuan Li, Piji Li, Quanzhi Li, Shaohua Li, Sheng Li, Shuangyin Li, Si Li, Sujian Li, Tao Li, Wei Li, Weikang Li, Wenjie Li, Xiang Li, Xiang Li, Xiaoli Li, Xin Li, Xintong Li, Xiujun Li, Yanyang Li, Yitong Li, Yuan-Fang Li, Yuncong Li, Zhenghua Li, Zhi Li, Zhixu Li, Zichao Li, Zuchao Li, Chao-Chun Liang, Paul Pu Liang, Hao Liao, Mark Liberman, Jindřich Libovický, Mohamed Lichouri, Chaya Liebeskind, Gilbert Lim, Bill Yuchen Lin, Chia-Wen Lin, Chuan-Jie Lin, Lucy H. Lin, Shou-de Lin, Xi Victoria Lin, Ye Lin, Zhouhan Lin, Zi Lin, Yuan Ling, Zhenhua Ling, Aldo Lipani, Tom Lippincott, Pierre Lison, Bang Liu, Bing Liu, Chenxi Liu, Dexi Liu, Fei Liu, Fenglin Liu, Gongshen Liu, Guiliang Liu, Han Liu, Haochen Liu, Jiangming Liu, Jing Liu, Kang Liu, Lemao Liu, Maofu Liu, Ming Liu, Ming Liu, Nelson F. Liu, Peng Liu, Qun Liu, Shuang Liu, Shujie Liu, Sijia Liu, Tianyi Liu, Tianyu Liu, Ting Liu, Tong Liu, Xianggen Liu, Xiaodong Liu, Xiaoyuan Liu, Xuanqing Liu, Yang Liu, Yijia Liu, Yiqun Liu, Yong Liu, Yongbin Liu, Zhenghao Liu, Zhibin Liu, Zhiyuan Liu, Zhiyuan Liu, Zhuang Liu, Zitao Liu, Alexander Loeser, Usha Lokala, Congjun Long, José Lopes, Marcos Lopes, Henrique Lopes Cardoso, Oier Lopez de Lacalle, Jaime Lorenzo-Trueba, Jian-Guang LOU, Natalia Loukachevitch, Daniel Loureiro, Ismini Lourentzou, Kate Loveys, Dawei Lu, Di Lu, Jianguo Lu, Jing Lu, Wei Lu, Weiming Lu, Yanbin Lu, Yichao Lu, Gale Lucas, Haozheng LUO, Liangchen Luo, Ling Luo, Weihua Luo, Wencan Luo, Zhunchen Luo, Chunchuan Lyu, Yajuan Lyu

Jianqiang Ma, Jing Ma, Long-Long Ma, Wei-Yun Ma, Xuezhe Ma, Aman Madaan, Andrea Madotto, Manuel Mager, Diwakar Mahajan, Debanjan Mahata, Jean Maillard, Peter Makarov, Márton Makrai, Andreas Maletti, Igor Malivoutov, Valentin Malykh, Radhika

Mamidi, Saab Mansour, Ramesh Manuvinakurike, Jiaxin Mao, Xian-Ling Mao, Ana Marasović, Daniel Marcu, Benjamin Marie, Alex Marin, Edison Marrese-Taylor, Bruno Martins, David Martins de Matos, Luis Martí, Héctor Martínez Alonso, Eva Martínez Garcia, Eugenio Martínez-Cámarra, Pascual Martínez-Gómez, Luis Marujo, Prashant Mathur, Sérgio Matos, Takuya Matsuzaki, Yevgen Matusevych, Abhinav Maurya, Jonathan May, Sahisnu Mazumder, Arya D. McCarthy, David McClosky, John Philip McCrae, Stephen McGregor, Bridget McInnes, Yashar Mehdad, Sachin Mehta, Hongyuan Mei, Arul Menezes, Fan-chao Meng, Rui Meng, Kourosh Meshgi, Lars Meyer, Ivan Vladimir Meza Ruiz, Meryem M'hamdi, Haitao Mi, Julian Michael, Stuart Middleton, Sabrina J. Mielke, Margot Mieskes, Claudiu Mihăilă, Elena Mikhalkova, Simon Mille, Timothy Miller, Erxue Min, Qingkai Min, Sewon Min, Koji Mineshima, Sabino Miranda-Jiménez, Seyed Abolghasem Mirroshandel, Paramita Mirza, Azadeh Mirzaei, Abhijit Mishra, Shubhanshu Mishra, Amita Misra, Jelena Mitrović, Sudip Mittal, Makoto Miwa, Junta Mizuno, Daichi Mochihashi, Ashutosh Modi, Aditya Mogadala, Omid Mohamad Nezami, Alaa Mohasseb, Michael Mohler, Diego Molla, Nicholas Monath, Maria Montefinese, Manuel Montes, Lori Moon, Taesun Moon, Nafise Sadat Moosavi, Richard Moot, Junichiro Mori, Emmanuel Morin, Lawrence Moss, Lili Mou, Ahmed Mourad, Diego Moussalem, Jiaqi Mu, Pramod Kaushik Mudrakarta, Animesh Mukherjee, Matthew Mulholland, Varish Mulwad, Emir Munoz, Dragos Munteanu, Mark-Christoph Müller

Seung-Hoon Na, Farah Nadeem, Maria Nadejde, Seema Nagar, Masaaki Nagata, Ajay Nagesh, Seiichi Nakagawa, Tetsuji Nakagawa, Toshiaki Nakazawa, Preslav Nakov, Diane Napolitano, Jason Naradowsky, Sharan Narasimhan, Sudip Kumar Naskar, Alexis Nasr, Vivi Nastase, Prem Natarajan, Roberto Navigli, Hamada Nayel, Adeline Nazarenko, Matteo Negri, Mariana Neves, Denis Newman-Griffis, Vincent Ng, Axel-Cyrille Ngonga Ngomo, Dat Quoc Nguyen, Huy Nguyen, Kim Anh Nguyen, Nhung Nguyen, Thien Huu Nguyen, Truc-Vien T. Nguyen, Eric Nichols, Garrett Nicolai, Massimo Nicosia, Vlad Niculae, Jan Niehues, Alexander Nikitin, Giannis Nikolentzos, Qiang Ning, Takashi Ninomiya, Kyosuke Nishida, Masaaki Nishino, Sergiu Nisioi, Tong Niu, Xing Niu, Zheng-Yu Niu, Hiroshi Noji, Debora Nozza, Pierre Nugues

Tim Oates, Daniela Alejandra Ochoa, Alexander O'Connor, Yusuke Oda, Stephan Oepen, Kemal Oflazer, Tim O'Gorman, Maciej Ograniczuk, Alice Oh, Jong-Hoon Oh, Tomoko Ohkuma, Kiyonori Ohtake, Atul Kr. Ojha, Naoaki Okazaki, Tsuyoshi Okita, Manabu Okumura, Oleg Okun, Antoni Oliver, Olabiyi Oluwatobi, Ethel Ong, Shereen Oraby, Constantin Orasan, Maite Oronoz, Farhad Oroumchian, Naoki Otani, Myle Ott, Hiroki Ouchi, Jessica Ouyang

Deepak P, Veno Pachovski, Ankur Padia, Ulrike Pado, Patrizia Paggio, Santanu Pal, Shruti Palaskar, Martha Palmer, Yi-Cheng Pan, Liang Pang, Manos Papagelis, Haris Papageorgiou, Alexandros Papangelis, Nikos Papasarantopoulos, Nikolaos Pappas, Ivandré Paraboni, Emerson Paraiso, Natalie Parde, Antonio Pareja-Lora, Shantipriya Parida, Eunjeong Park, Kunwoo Park, Ioannis Partalas, Tommaso Pasini, Marco Passarotti, Rebecca J. Passonneau, Ramakanth Pasunuru, Roma Patel, Braja Gopal Patra, Ellie Pavlick, Pavel Pecina, Stephan Peitz, Hao Peng, Haoruo Peng, Minlong Peng, Wei Peng, Xingchao Peng, Yifan Peng, Zixuan Peng, Gerald Penn, Lis Pereira, Gabriele Pergola, Matthew E. Peters, Maxime Peyrard, Sandro Pezzelle, Anselmo Peñas, Quang Nhat Minh Pham, Tommi Pirinen, Nikiforos Pititaras, Lidia Pivovarova, Emmanouil Antonios Platanios, Livia Polanyi, Edoardo Maria Ponti, Simone Paolo Ponzetto, Hanieh Poostchi, Lucian Popa, Octavian Popescu, Maja Popović, Fred Popowich, Christopher Potts, Pascal Poupart, Sandhya Prabhakaran, Animesh Prasad, Daniel Preotiuc-Pietro, Emily Prud'hommeaux, Stephen Pulman, Matthew Purver

Vahed Qazvinian, Jianzhong Qi, Peng Qi, Chen Qian, Tieyun Qian, Yujie Qian, Zhou Qin, Xinying Qiu, Xipeng Qiu, Chen Qu, Lizhen Qu

Joanna Rabiega-Wisniewska, Ella Rabinovich, Alexandre Rademaker, Afshin Rahimi, Muhammad Rahman, Sunny Rai, Dheeraj Rajagopal, Nitendra Rajput, Dhananjay Ram, Taraka Rama, Deepak Ramachandran, Lakshmi Ramachandran, Rohan Ramanath, Gabriela Ramirez-de-la-Rosa, Carlos Ramisch, Nitin Ramrakhiyani, Surangika Ranathunga, Vivek Kumar Rangarajan Sridhar, Yanghui Rao, Ari Rappoport, Mohammad Sadegh Rasooli, Abhinav Rastogi, Sujith Ravi, Manikandan Ravikiran, Vinit Ravishankar, Livy Real, Traian Rebedea, Hanumant Redkar, Ines Rehbein, Georg Rehm, Julia Reinspach, Navid Rekabsaz, Steffen Remus, han ren, Yafeng Ren, Zhaochun Ren, Corentin Ribeyre, German Rigau, Tharathorn Rimchala, Laura Rimell, Annette Rios, Anthony Rios, Gil Rocha, Paul Rodrigues, Lina M. Rojas Barahona, Laurent Romary, Srikanth Ronanki, Wenge Rong, Rudolf Rosa, Andrew Rosenberg, Candace Ross, Sophie Rosset, Michael Roth, Masoud Rouhizadeh, Alla Rozovskaya, Sebastian Ruder, Frank Rudzicz, Nicholas Ruiz, Josef Ruppenhofer, Irene Russo, Attapol Rutherford, Rafal Rzepka

Devendra Sachan, Kugatsu Sadamitsu, Fatiha Sadat, Pegah Safari, Sylvie Saget, Koustuv Saha, Monjoy Saha, Rishiraj Saha Roy, Saurav Sahay, Gözde Gül Şahin, Magnus Sahlgren, Sunil Kumar Sahu, Hassan Sajjad, Keisuke Sakaguchi, Sakriani Sakti, Mohammad Salameh, Iman Saleh, Elizabeth Salesky, Avneesh Saluja, Tanja Samardzic, Germán Sanchis-Trilles, Hugo Sanjurjo González, Ananth Sankar, Praba Santhanakrishnan, Diana Santos, Enrico Santus, Soumya Sanyal, Kamal Sarkar, Felix Sasaki, Shota Sasaki, Bianca Scarlini, Carolina Scarton, Shigehiko Schamoni, David Schlangen, Dominik Schlechtweg, Natalie Schluter, Helmut Schmid, Sylvain Schmitz, Jodi Schneider, William Schuler, Sabine Schulte im Walde, Sebastian Schuster, Satoshi Sekine, Ethan Selfridge, David Semedo, Diarmuid Ó Séaghdha, Nasredine Semmar, Gregory Senay, Rico Sennrich, Minjoon Seo, Christophe Servan, Lei Sha, Pararth Shah, Samira Shaikh, Cory Shain, Azadeh Shakery, Jingbo Shang, Mingyue Shang, Jie Shao, Ori Shapira, Piyush Sharma, Raksha Sharma, Soumya Sharma, Serge Sharoff, Zaid Sheikh, RAVI SHEKHAR, Artem Shelmanov, Wade Shen, Yilin Shen, Yuming Shen, Hao Sheng, Michael Sheng, Bei Shi, Chongyang Shi, Peng Shi, Tian Shi, Tianze Shi, Wei Shi, Wenxian Shi, Xing Shi, Yangyang Shi, Tomohide Shibata, Yutaro Shigeto, Takahiro Shinozaki, Chaitanya Shivade, Lidan Shou, Manish Shrivastava, Dimitar Shterionov, Kai Shu, Lei Shu, Kai Shuang, Maryam Siahbani, Aditya Siddhant, Diego Silva, João Silva, Fabrizio Silvestri, Stefano Silvestri, Michel Simard, Patrick Simianer, Marian Simko, Dan Simonson, Edwin Simpson, Abhishek Singh, Mayank Singh, Rajeev Kumar Singh, Karan Singla, Amando Jr. Singun, Olivier siohan, Amy Siu, Marco Antonio Sobrevilla Cabezudo, Luca Soldaini, Dongjin Song, Hyun-Je Song, Linfeng Song, Wei Song, Xingyi Song, Yan Song, YIPING SONG, Yuxuan Song, Radu Soricu, Alexey Sorokin, Daniil Sorokin, Marlo Souza, Manuela Speranza, Damiano Spina, Balaji Vasan Srinivasan, Edward Stabler, Felix Stahlberg, Efstatios Stamatatos, Miloš Stanojević, Mark Steedman, Shane Steinert-Threlkeld, Zachary Stine, Kurt Stockinger, Michael Strube, Tomek Strzalkowski, Sara Stymne, Jinsong Su, Keh-Yih Su, Ming-Hsiang Su, Qinliang Su, Shang-Yu Su, Weifeng Su, Aparna Subramanian, Katsuhito Sudoh, Kazunari Sugiyama, Derwin Suhartono, Alane Suhr, Chengjie Sun, Le Sun, Lin Sun, Ming Sun, Weiwei Sun, Yawei Sun, Yibo Sun, Kalaivani Sundararajan, Hanna Suominen, Jun Suzuki, Yoshimi Suzuki, Ida Szubert, Joan Andreu Sánchez

Jeniya Tabassum, Ryuki Tachibana, Kaveh Taghipour, Nina Tahmasebi, Sho Takase, David Talbot, Yik-Cheung Tam, George Tambouratzis, Akihiro Tamura, FEI TAN, Hao Tan, Jiwei

Tan, Ming Tan, Kumiko Tanaka-Ishii, Duyu Tang, Gongbo Tang, Jie Tang, Qingming Tang, Raphael Tang, Shuai Tang, Siliang Tang, Xiangyun Tang, Chongyang Tao, Fei Tao, Yifeng Tao, Yuka Tateisi, Marta Tatu, Yi Tay, Andon Tchechmedjiev, Christoph Teichmann, Selma Tekir, Irina Temnikova, Zhiyang Teng, Joel Tetreault, Uthayasanker Thayasivam, Krishnaprasad Thirunarayan, Jesse Thomason, Brian Thompson, Camilo Thorne, Veronika Thost, Yuan Tian, Swati Tiwari, Erik Tjong Kim Sang, Takenobu Tokunaga, Gaurav Singh Tomar, Nadi Tomeh, Mariya Toneva, Kentaro Torisawa, Adrià Torrens Urrutia, Samia Touileb, Ke Tran, Quan Hung Tran, Trang Tran, Diana Trandabat, Alina Trifan, Jan Trmal, Chen-Tse Tsai, Adam Tsakalidis, Yuen-Hsien Tseng, Yulia Tsvetkov, Lifu Tu, Ming Tu, Zhaopeng Tu, Aaron Tuor, Gokhan Tur, Marco Turchi, Ferhan Ture, Rory Turnbull, Martin Tutek

Kiyotaka Uchimoto, Adrian Ulges, Lyle Ungar, Shyam Upadhyay, L. Alfonso Urena Lopez, Zdenka Uresova, Masao Utiyama, Takehito Utsuro

Sowmya Vajjala, Tim Van de Cruys, Rob van der Goot, Keith VanderLinden, David Vandyke, Natalia Vanetik, Clara Vania, Andrea Varga, Shikhar Vashishth, Alakananda Vempala, Rakesh Verma, Guido Vetere, David Vilares, Jesús Vilares, Jesus Villalba, Serena Villata, Esau Villatoro-Tello, Aline Villavicencio, Anne Vilnat, Veronika Vincze, Jacky Visser, Rob Voigt, Soroush Vosoughi, Ngoc Thang Vu, Thuy Vu, Ivan Vulić, Ekaterina Vylomova

Henning Wachsmuth, Ada Wan, Mengting Wan, Stephen Wan, Xiaojun Wan, Ante Wang, Baoxun Wang, Chao Wang, Cheng Wang, Chuan-Ju Wang, Cunxiang Wang, Daling Wang, DERUI WANG, Di Wang, Dingquan Wang, Fu Lee Wang, Guangrun Wang, Guangtao Wang, Guoyin Wang, Hai Wang, Han Wang, Hao Wang, Hao Wang, Haohan Wang, Hong Wang, Hsin-Min Wang, Jiang Wang, Jianzong Wang, Jin Wang, Jingjing Wang, Jun Wang, Jun Wang, Junfeng Wang, Ke Wang, Lei Wang, Lijun Wang, Liwei Wang, Longyue Wang, Lu Wang, Nan Wang, Pidong Wang, Ping Wang, Qingyun Wang, Quan Wang, Rui Wang, Rui Wang, Shuai Wang, Shuohang Wang, Shuting Wang, Tong Wang, Wei Wang, Wei Wang, Weiyue Wang, Wenqi Wang, Xiang Wang, Xiaojie WANG, Xiaolin Wang, Xin Wang, Xing Wang, Xintong Wang, Xinyu Wang, Xuancong Wang, Xuezhi Wang, Yan Wang, Yanshan Wang, Yiou Wang, Yizhong Wang, Yue Wang, Zheng Wang, Zhiguang Wang, Zhongqing Wang, Artit Wangperawong, Leo Wanner, Nigel Ward, Zeerak Waseem, Taro Watanabe, Roger Wattenhofer, Andy Way, Bonnie Webber, Ingmar Weber, Feng Wei, Jason Wei, Wei Wei, Xiangpeng Wei, Zhongyu Wei, Charles Welch, Simon Wells, Aaron Steven White, John Wieting, Derry Tanti Wijaya, Gijs Wijnholds, Rodrigo Wilkens, Jennifer Williams, Grégoire Winterstein, Magdalena Wolska, Derek F. Wong, Kam-Fai Wong, Tak-Lam Wong, Dina Wonsever, Alina Wróblewska, Changxing Wu, Chien-Sheng Wu, Fangzhao Wu, Guani Wu, Hao Wu, Hua Wu, Ji Wu, Jian Wu, Kui Wu, Lijun Wu, Lingfei Wu, Mengyue Wu, Ou Wu, Stephen Wu, Xianchao Wu, Xing Wu, Youzheng Wu, Yuanbin Wu, Zhiyong Wu, Joern Ruebker

Aris Xanthos, Xuefeng Xi, Yikun Xian, Chunyang Xiao, Han Xiao, Huiru Xiao, MIN XIAO, Tong Xiao, Xinyan Xiao, Yanghua Xiao, Boyi Xie, Qizhe Xie, Ruobing Xie, Yingwei Xin, Frank Xing, Yujie Xing, Zhenchang Xing, Chao Xiong, Deyi Xiong, hao xiong, Hongyu Xiong, Bo Xu, Canwen Xu, Dongkuan Xu, Fan XU, Guandong Xu, Hainan Xu, Hongzhi Xu, Hu Xu, Hua Xu, Jia Xu, jiaming xu, Jinan Xu, Kun Xu, Qiongkai Xu, Ruifeng Xu, Tong Xu, Weiran XU, Wenduan Xu, Yang Xu, Nianwen Xue, Qinghan Xue

Shuntaro Yada, Yadollah Yaghoobzadeh, Ikuya Yamada, Takehiro Yamamoto, Ming Yan, Xiaohui Yan, Baosong Yang, Bishan Yang, Eugene Yang, Haiqin Yang, Jie Yang, Jie Yang, Jingxuan Yang, Liner Yang, Liu Yang, Longfei Yang, Min Yang, Qian Yang, Shaohua Yang,

Wei Yang, Yaqin Yang, Yujiu Yang, Ze Yang, Zhenglu Yang, Zhengyuan Yang, Zi Yang, Zixiaofan Yang, T Yano, Jianmin YAO, Jin-Ge Yao, Kaisheng Yao, Liang Yao, Wenlin Yao, Ziyu Yao, Hai Ye, Reyyan Yeniterzi, Wen-tau Yih, Seid Yimam, Qingyu Yin, Seunghyun Yoon, Masaharu Yoshioka, Dian Yu, Dong Yu, Heng Yu, Hong Yu, Jianfei Yu, Jing Yu, Liang-Chih Yu, Tao Yu, Nicholas Jing Yuan, Zhaoquan Yuan, Chuan Yue, Frances Yung

Marcos Zampieri, Fadi Zaraket, Gian Piero ZARRI, Omnia Zayed, Yury Zemlyanskiy, Daojian Zeng, Xingshan Zeng, Zhaohao Zeng, Torsten Zesch, Deniz Zeyrek, Sheng Zha, Feifei Zhai, Shuang (Sophie) Zhai, Biao Zhang, Boliang Zhang, Chao Zhang, Cheng Zhang, Chengzhi Zhang, Chenwei Zhang, Dongdong Zhang, Dongyu Zhang, Guangwei Zhang, Hongming Zhang, Huangpan Zhang, Jiajun Zhang, Jing Zhang, Jinnian Zhang, Jipeng Zhang, Lei Zhang, Meishan Zhang, Min Zhang, Ming Zhang, Ningyu Zhang, Qi Zhang, Richong Zhang, Ruiyi Zhang, Ruqing Zhang, Sheng Zhang, Shuai Zhang, Wei Zhang, Wei-Nan Zhang, Wen Zhang, Wen Zhang, Xiangliang Zhang, Xiaojun Zhang, Xiaotong Zhang, Xuan Zhang, Yazhou Zhang, Yi Zhang, Yi Zhang, Yin Zhang, Yin Zhang, Yin Zhang, Yingyi Zhang, Yongfei Zhang, Yongfeng Zhang, Yuchen Zhang, Yue Zhang, Yue Zhang, Yuqi Zhang, Yuxiang Zhang, Yuyu Zhang, Zhe Zhang, Zhirui Zhang, Zhuosheng Zhang, Bing Zhao, Dongyan Zhao, Hai Zhao, Kai Zhao, Lin Zhao, Ruihui Zhao, Sendong Zhao, Shu Zhao, Tiancheng Zhao, Tiejun Zhao, Wayne Xin Zhao, Xiang Zhao, Xiaobing Zhao, Zhengli Zhao, Baigong Zheng, Renjie Zheng, Xiaoqing Zheng, Xin Zheng, Zaixiang Zheng, Alisa Zhila, Zexuan Zhong, Bin Zhou, Dong Zhou, Guangyou Zhou, Guorui Zhou, Jie Zhou, Jingbo Zhou, Joey Tianyi Zhou, Nina Zhou, Qiang Zhou, Qingyu Zhou, Wenxuan Zhou, Xiaobing Zhou, Yang Zhou, Dongxiao Zhu, Hao Zhu, Hengshu Zhu, Kenny Zhu, Linchao Zhu, Muhua Zhu, Su Zhu, Wentao Zhu, Leonardo Zilio, Heike Zinsmeister, Imed Zitouni, Michael Zock, Bowei Zou, Yuexian Zou, Arkaitz Zubiaga, Frederike Zufall



## Table of Contents

<i>Touch Editing: A Flexible One-Time Interaction Approach for Translation</i>	
Qian Wang, Jiajun Zhang, Lemao Liu, Guoping Huang and Chengqing Zong .....	1
<i>Can Monolingual Pretrained Models Help Cross-Lingual Classification?</i>	
Zewen Chi, Li Dong, Furu Wei, Xian-Ling Mao and Heyan Huang .....	12
<i>Rumor Detection on Twitter Using Multiloss Hierarchical BiLSTM with an Attenuation Factor</i>	
Yudianto Sujana, Jiawen Li and Hung-Yu Kao .....	18
<i>Graph Attention Network with Memory Fusion for Aspect-level Sentiment Analysis</i>	
Li Yuan, Jin Wang, Liang-Chih Yu and Xuejie Zhang .....	27
<i>FERNet: Fine-grained Extraction and Reasoning Network for Emotion Recognition in Dialogues</i>	
Yingmei Guo, Zhiyong Wu and Mingxing Xu .....	37
<i>SentiRec: Sentiment Diversity-aware Neural News Recommendation</i>	
Chuhan Wu, Fangzhao Wu, Tao Qi and Yongfeng Huang .....	44
<i>BCTH: A Novel Text Hashing Approach via Bayesian Clustering</i>	
Ying Wenjie, Yuquan Le and Hantao Xiong .....	54
<i>Lightweight Text Classifier using Sinusoidal Positional Encoding</i>	
Byoung-Doo Oh and Yu-Seop Kim .....	63
<i>Towards Non-task-specific Distillation of BERT via Sentence Representation Approximation</i>	
Bowen Wu, Huan Zhang, MengYuan Li, Zongsheng Wang, Qihang Feng, Junhong Huang and Baoxun Wang .....	70
<i>A Simple and Effective Usage of Word Clusters for CBOW Model</i>	
Yukun Feng, Chenlong Hu, Hidetaka Kamigaito, Hiroya Takamura and Manabu Okumura .....	80
<i>Investigating Learning Dynamics of BERT Fine-Tuning</i>	
Yaru Hao, Li Dong, Furu Wei and Ke Xu .....	87
<i>Second-Order Neural Dependency Parsing with Message Passing and End-to-End Training</i>	
Xinyu Wang and Kewei Tu .....	93
<i>High-order Refining for End-to-end Chinese Semantic Role Labeling</i>	
Hao Fei, Yafeng Ren and Donghong Ji .....	100
<i>Exploiting WordNet Synset and Hypernym Representations for Answer Selection</i>	
Weikang Li and Yunfang Wu .....	106
<i>A Simple Text-based Relevant Location Prediction Method using Knowledge Base</i>	
Mei Sasaki, Shumpei Okura and Shingo Ono .....	116
<i>Learning Goal-oriented Dialogue Policy with opposite Agent Awareness</i>	
Zheng Zhang, Lizi Liao, Xiaoyan Zhu, Tat-Seng Chua, Zitao Liu, Yan Huang and Minlie Huang .....	122
<i>An Empirical Study of Tokenization Strategies for Various Korean NLP Tasks</i>	
Kyubyong Park, Joohong Lee, Seongbo Jang and Dawoon Jung .....	133

<i>BERT-Based Neural Collaborative Filtering and Fixed-Length Contiguous Tokens Explanation</i>	Reinald Adrian Pugoy and Hung-Yu Kao .....	143
<i>Transformer-based Approach for Predicting Chemical Compound Structures</i>	Yutaro Omote, Kyoumoto Matsushita, Tomoya Iwakura, Akihiro Tamura and Takashi Ninomiya	154
<i>Chinese Grammatical Correction Using BERT-based Pre-trained Model</i>	Hongfei Wang, Michiki Kurosawa, Satoru Katsumata and Mamoru Komachi .....	163
<i>Neural Gibbs Sampling for Joint Event Argument Extraction</i>	Xiaozhi Wang, Shengyu Jia, Xu Han, Zhiyuan Liu, Juanzi Li, Peng Li and Jie Zhou .....	169
<i>Named Entity Recognition in Multi-level Contexts</i>	Yubo Chen, Chuhan Wu, Tao Qi, Zhigang Yuan and Yongfeng Huang.....	181
<i>A General Framework for Adaptation of Neural Machine Translation to Simultaneous Translation</i>	Yun Chen, Liangyou Li, Xin Jiang, Xiao Chen and Qun Liu.....	191
<i>UnihanLM: Coarse-to-Fine Chinese-Japanese Language Model Pretraining with the Unihan Database</i>	Canwen Xu, Tao Ge, Chenliang Li and Furu Wei .....	201
<i>Towards a Better Understanding of Label Smoothing in Neural Machine Translation</i>	Yingbo Gao, Weiyue Wang, Christian Herold, Zijian Yang and Hermann Ney.....	212
<i>Comparing Probabilistic, Distributional and Transformer-Based Models on Logical Metonymy Interpretation</i>	Giulia Rambelli, Emmanuele Chersoni, Alessandro Lenci, Philippe Blache and Chu-Ren Huang	224
<i>AMR Quality Rating with a Lightweight CNN</i>	Juri Opitz .....	235
<i>Generating Commonsense Explanation by Extracting Bridge Concepts from Reasoning Paths</i>	Haozhe Ji, Pei Ke, Shaohan Huang, Furu Wei and Minlie Huang .....	248
<i>Unsupervised KB-to-Text Generation with Auxiliary Triple Extraction using Dual Learning</i>	Zihao Fu, Bei Shi, Lidong Bing and Wai Lam .....	258
<i>Modality-Transferable Emotion Embeddings for Low-Resource Multimodal Emotion Recognition</i>	Wenliang Dai, Zihan Liu, Tiezheng Yu and Pascale Fung .....	269
<i>All-in-One: A Deep Attentive Multi-task Learning Framework for Humour, Sarcasm, Offensive, Motivation, and Sentiment on Memes</i>	Dushyant Singh Chauhan, Dhanush S R, Asif Ekbal and Pushpak Bhattacharyya .....	281
<i>Identifying Implicit Quotes for Unsupervised Extractive Summarization of Conversations</i>	Ryuji Kano, Yasuhide Miura, Tomoki Taniguchi and Tomoko Ohkuma .....	291
<i>Unsupervised Aspect-Level Sentiment Controllable Style Transfer</i>	Mukuntha Narayanan Sundararaman, Zishan Ahmad, Asif Ekbal and Pushpak Bhattacharyya .	303
<i>Energy-based Self-attentive Learning of Abstractive Communities for Spoken Language Understanding</i>	Guokan Shang, Antoine Tixier, Michalis Vazirgiannis and Jean-Pierre Lorré .....	313

<i>Intent Detection with WikiHow</i>	
Li Zhang, Qing Lyu and Chris Callison-Burch .....	328
<i>A Systematic Characterization of Sampling Algorithms for Open-ended Language Generation</i>	
Moin Nadeem, Tianxing He, Kyunghyun Cho and James Glass.....	334
<i>Chinese Content Scoring: Open-Access Datasets and Features on Different Segmentation Levels</i>	
Yuning Ding, Andrea Horbach and Torsten Zesch .....	347
<i>Analysis of Hierarchical Multi-Content Text Classification Model on B-SHARP Dataset for Early Detection of Alzheimer's Disease</i>	
Renxuan Albert Li, Ihab Hajjar, Felicia Goldstein and Jinho D. Choi .....	358
<i>An Exploratory Study on Multilingual Quality Estimation</i>	
Shuo Sun, Marina Fomicheva, Frédéric Blain, Vishrav Chaudhary, Ahmed El-Kishky, Adithya Renduchintala, Francisco Guzmán and Lucia Specia .....	366
<i>English-to-Chinese Transliteration with Phonetic Auxiliary Task</i>	
Yuan He and Shay B. Cohen .....	378
<i>Predicting and Using Target Length in Neural Machine Translation</i>	
Zijian Yang, Yingbo Gao, Weiyue Wang and Hermann Ney .....	389
<i>Grounded PCFG Induction with Images</i>	
Lifeng Jin and William Schuler .....	396
<i>Heads-up! Unsupervised Constituency Parsing via Self-Attention Heads</i>	
Bowen Li, Taeuk Kim, Reinald Kim Amplayo and Frank Keller .....	409
<i>Building Location Embeddings from Physical Trajectories and Textual Representations</i>	
Laura Biester, Carmen Banea and Rada Mihalcea .....	425
<i>Self-Supervised Learning for Pairwise Data Refinement</i>	
Gustavo Hernandez Abrego, Bowen Liang, Wei Wang, Zarana Parekh, Yinfei Yang and Yunhsuan Sung .....	435
<i>A Survey of the State of Explainable AI for Natural Language Processing</i>	
Marina Danilevsky, Kun Qian, Ranit Aharonov, Yannis Katsis, Ban Kawas and Prithviraj Sen .	447
<i>Beyond Fine-tuning: Few-Sample Sentence Embedding Transfer</i>	
Siddhant Garg, Rohit Kumar Sharma and Yingyu Liang .....	460
<i>Multimodal Pretraining for Dense Video Captioning</i>	
Gabriel Huang, Bo Pang, Zhenhai Zhu, Clara Rivera and Radu Soricut .....	470
<i>Systematic Generalization on gSCAN with Language Conditioned Embedding</i>	
Tong Gao, Qi Huang and Raymond Mooney.....	491
<i>Are Scene Graphs Good Enough to Improve Image Captioning?</i>	
Victor Milewski, Marie-Francine Moens and Iacer Calixto .....	504
<i>Systematically Exploring Redundancy Reduction in Summarizing Long Documents</i>	
Wen Xiao and Giuseppe Carenini .....	516
<i>A Cascade Approach to Neural Abstractive Summarization with Content Selection and Fusion</i>	
Logan Lebanoff, Franck Dernoncourt, Doo Soon Kim, Walter Chang and Fei Liu .....	529

<i>Mixed-Lingual Pre-training for Cross-lingual Summarization</i>	
Ruochen Xu, Chenguang Zhu, Yu Shi, Michael Zeng and Xuedong Huang .....	536
<i>Point-of-Interest Oriented Question Answering with Joint Inference of Semantic Matching and Distance Correlation</i>	
Yifei Yuan, Jingbo Zhou and Wai Lam .....	542
<i>Leveraging Structured Metadata for Improving Question Answering on the Web</i>	
Xinya Du, Ahmed Hassan Awadallah, Adamour Fournier, Robert Sim, Paul Bennett and Claire Cardie .....	546
<i>English Intermediate-Task Training Improves Zero-Shot Cross-Lingual Transfer Too</i>	
Jason Phang, Iacer Calixto, Phu Mon Htut, Yada Pruksachatkun, Haokun Liu, Clara Vania, Katharina Kann and Samuel R. Bowman .....	557
<i>STIL - Simultaneous Slot Filling, Translation, Intent Classification, and Language Identification: Initial Results using mBART on MultiATIS++</i>	
Jack FitzGerald .....	576
<i>SimulMT to SimulST: Adapting Simultaneous Text Translation to End-to-End Simultaneous Speech Translation</i>	
Xutai Ma, Juan Pino and Philipp Koehn .....	582
<i>Cue Me In: Content-Inducing Approaches to Interactive Story Generation</i>	
Faeze Brahman, Alexandru Petrusca and Snigdha Chaturvedi .....	588
<i>Liputan6: A Large-scale Indonesian Dataset for Text Summarization</i>	
Fajri Koto, Jey Han Lau and Timothy Baldwin .....	598
<i>Generating Sports News from Live Commentary: A Chinese Dataset for Sports Game Summarization</i>	
Kuan-Hao Huang, Chen Li and Kai-Wei Chang .....	609
<i>Massively Multilingual Document Alignment with Cross-lingual Sentence-Mover's Distance</i>	
Ahmed El-Kishky and Francisco Guzmán .....	616
<i>Improving Context Modeling in Neural Topic Segmentation</i>	
Linzi Xing, Brad Hackinen, Giuseppe Carenini and Francesco Trebbi .....	626
<i>Contextualized End-to-End Neural Entity Linking</i>	
Haotian Chen, Xi Li, Andrej Zukov Gregoric and Sahil Wadhwa .....	637
<i>DAPPER: Learning Domain-Adapted Persona Representation Using Pretrained BERT and External Memory</i>	
Prashanth Vijayaraghavan, Eric Chu and Deb Roy .....	643
<i>Event Coreference Resolution with Non-Local Information</i>	
Jing Lu and Vincent Ng .....	653
<i>Neural RST-based Evaluation of Discourse Coherence</i>	
Grigorii Guz, Peyman Batani, Darius Muglich and Giuseppe Carenini .....	664
<i>Asking Crowdworkers to Write Entailment Examples: The Best of Bad Options</i>	
Clara Vania, Ruijie Chen and Samuel R. Bowman .....	672

<i>MaP: A Matrix-based Prediction Approach to Improve Span Extraction in Machine Reading Comprehension</i>	
Huaishao Luo, Yu Shi, Ming Gong, Linjun Shou and Tianrui Li .....	687
<i>Answering Product-related Questions with Heterogeneous Information</i>	
Wenxuan Zhang, Qian Yu and Wai Lam.....	696
<i>Two-Step Classification using Recasted Data for Low Resource Settings</i>	
Shagun Uppal, Vivek Gupta, Avinash Swaminathan, Haimin Zhang, Debanjan Mahata, Rakesh Gosangi, Rajiv Ratn Shah and Amanda Stent.....	706
<i>Explaining Word Embeddings via Disentangled Representation</i>	
Keng-Te Liao, Cheng-Syuan Lee, Zhong-Yu Huang and Shou-de Lin .....	720
<i>Multi-view Classification Model for Knowledge Graph Completion</i>	
Wenbin Jiang, Mengfei Guo, Yufeng Chen, Ying Li, Jinan Xu, Yajuan Lyu and Yong Zhu.....	726
<i>Knowledge-Enhanced Named Entity Disambiguation for Short Text</i>	
Zhifan Feng, Qi Wang, Wenbin Jiang, Yajuan Lyu and Yong Zhu.....	735
<i>More Data, More Relations, More Context and More Openness: A Review and Outlook for Relation Extraction</i>	
Xu Han, Tianyu Gao, Yankai Lin, Hao Peng, Yaoliang Yang, Chaojun Xiao, Zhiyuan Liu, Peng Li, Jie Zhou and Maosong Sun.....	745
<i>Robustness and Reliability of Gender Bias Assessment in Word Embeddings: The Role of Base Pairs</i>	
Haiyang Zhang, Alison Sneyd and Mark Stevenson .....	759
<i>ExpanRL: Hierarchical Reinforcement Learning for Course Concept Expansion in MOOCs</i>	
Jifan Yu, Chenyu Wang, Gan Luo, Lei Hou, Juanzi Li, Jie Tang, Minlie Huang and Zhiyuan Liu	770
<i>Vocabulary Matters: A Simple yet Effective Approach to Paragraph-level Question Generation</i>	
Vishwajeet Kumar, Manish Joshi, Ganesh Ramakrishnan and Yuan-Fang Li .....	781
<i>From Hero to Zéroé: A Benchmark of Low-Level Adversarial Attacks</i>	
Steffen Eger and Yannik Benz .....	786
<i>Point-of-Interest Type Inference from Social Media Text</i>	
Danae Sánchez Villegas, Daniel Preotiuc-Pietro and Nikolaos Aletras .....	804
<i>Reconstructing Event Regions for Event Extraction via Graph Attention Networks</i>	
Pei Chen, Hang Yang, Kang Liu, Ruihong Huang, Yubo Chen, Taifeng Wang and Jun Zhao ..	811
<i>Recipe Instruction Semantics Corpus (RISec): Resolving Semantic Structure and Zero Anaphora in Recipes</i>	
Yiwei Jiang, Klim Zaporojets, Johannes Deleu, Thomas Demeester and Chris Develder.....	821
<i>Stronger Baselines for Grammatical Error Correction Using a Pretrained Encoder-Decoder Model</i>	
Satoru Katsumata and Mamoru Komachi .....	827
<i>Sina Mandarin Alphabetical Words:A Web-driven Code-mixing Lexical Resource</i>	
Rong Xiang, Mingyu Wan, Qi Su, Chu-Ren Huang and Qin Lu.....	833
<i>IndoNLU: Benchmark and Resources for Evaluating Indonesian Natural Language Understanding</i>	
Bryan Wilie, Karissa Vincentio, Genta Winata, Samuel Cahyawijaya, Xiaohong Li, Zhi Yuan Lim, Sidik Soleman, Rahmad Mahendra, Pascale Fung, Syafri Bahar and Ayu Purwarianti.....	843

<i>Happy Are Those Who Grade without Seeing: A Multi-Task Learning Approach to Grade Essays Using Gaze Behaviour</i>	
Sandeep Mathias, Rudra Murthy, Diptesh Kanodia, Abhijit Mishra and Pushpak Bhattacharyya	858
<i>Multi-Source Attention for Unsupervised Domain Adaptation</i>	
Xia Cui and Danushka Bollegala .....	873
<i>Compressing Pre-trained Language Models by Matrix Decomposition</i>	
Matan Ben Noach and Yoav Goldberg .....	884
<i>You May Like This Hotel Because ...: Identifying Evidence for Explainable Recommendations</i>	
Shin Kanouchi, Masato Neishi, Yuta Hayashibe, Hiroki Ouchi and Naoaki Okazaki .....	890
<i>A Unified Framework for Multilingual and Code-Mixed Visual Question Answering</i>	
Deepak Gupta, Pabitra Lenka, Asif Ekbal and Pushpak Bhattacharyya .....	900
<i>Toxic Language Detection in Social Media for Brazilian Portuguese: New Dataset and Multilingual Analysis</i>	
João Augusto Leite, Diego Silva, Kalina Bontcheva and Carolina Scarton .....	914
<i>Measuring What Counts: The Case of Rumour Stance Classification</i>	
Carolina Scarton, Diego Silva and Kalina Bontcheva .....	925

# Conference Program

Saturday, Dec. 5 (all sessions are scheduled between 8:00 and 24:00 UTC +8)

**8:45–9:00** *Opening Remarks*

**9:00–10:00** *Keynote*

**10:00–11:00** **Session 1A: Machine Translation and Multilinguality I**

10:00–10:20 *Touch Editing: A Flexible One-Time Interaction Approach for Translation*  
Qian Wang, Jiajun Zhang, Lemao Liu, Guoping Huang and Chengqing Zong

10:20–10:35 *Can Monolingual Pretrained Models Help Cross-Lingual Classification?*  
Zewen Chi, Li Dong, Furu Wei, Xian-Ling Mao and Heyan Huang

**10:00–11:00** **Session 1B: Sentiment Analysis and Argument Mining I**

10:00–10:20 *Rumor Detection on Twitter Using Multiloss Hierarchical BiLSTM with an Attenuation Factor*  
Yudianto Sujana, Jiawen Li and Hung-Yu Kao

10:20–10:40 *Graph Attention Network with Memory Fusion for Aspect-level Sentiment Analysis*  
Li Yuan, Jin Wang, Liang-Chih Yu and Xuejie Zhang

10:40–10:55 *FERNet: Fine-grained Extraction and Reasoning Network for Emotion Recognition in Dialogues*  
Yingmei Guo, Zhiyong Wu and Mingxing Xu

**Saturday, Dec. 5 (all sessions are scheduled between 8:00 and 24:00 UTC +8) (continued)**

**10:00–11:00 Session 1C: Information Retrieval and Document Analysis I**

10:00–10:20 *SentiRec: Sentiment Diversity-aware Neural News Recommendation*

Chuhan Wu, Fangzhao Wu, Tao Qi and Yongfeng Huang

10:20–10:40 *BCTH: A Novel Text Hashing Approach via Bayesian Clustering*

Ying Wenjie, Yuquan Le and Hantao Xiong

10:40–10:55 *Lightweight Text Classifier using Sinusoidal Positional Encoding*

Byoung-Doo Oh and Yu-Seop Kim

**11:00–12:00 Session 2A: Machine Learning for NLP I**

11:00–11:20 *Towards Non-task-specific Distillation of BERT via Sentence Representation Approximation*

Bowen Wu, Huan Zhang, MengYuan Li, Zongsheng Wang, Qihang Feng, Junhong Huang and Baoxun Wang

11:20–11:35 *A Simple and Effective Usage of Word Clusters for CBOW Model*

Yukun Feng, Chenlong Hu, Hidetaka Kamigaito, Hiroya Takamura and Manabu Okumura

11:35–11:50 *Investigating Learning Dynamics of BERT Fine-Tuning*

Yaru Hao, Li Dong, Furu Wei and Ke Xu

**Saturday, Dec. 5 (all sessions are scheduled between 8:00 and 24:00 UTC +8) (continued)**

**11:00–12:00 Session 2B: Tagging, Chunking, Syntax, and Parsing I**

- 11:00–11:15 *Second-Order Neural Dependency Parsing with Message Passing and End-to-End Training*  
Xinyu Wang and Kewei Tu
- 11:15–11:30 *High-order Refining for End-to-end Chinese Semantic Role Labeling*  
Hao Fei, Yafeng Ren and Donghong Ji

**11:00–12:00 Session 2C: Knowledge Graph**

- 11:00–11:20 *Exploiting WordNet Synset and Hypernym Representations for Answer Selection*  
Weikang Li and Yunfang Wu
- 11:20–11:35 *A Simple Text-based Relevant Location Prediction Method using Knowledge Base*  
Mei Sasaki, Shumpei Okura and Shingo Ono

**12:00–14:00 Lunch Break**

- 14:00–15:00 Session 3A: Dialogue and Interactive Systems I & Phonology, Morphology and Word Segmentation I**
- 14:00–14:20 *Learning Goal-oriented Dialogue Policy with opposite Agent Awareness*  
Zheng Zhang, Lizi Liao, Xiaoyan Zhu, Tat-Seng Chua, Zitao Liu, Yan Huang and Minlie Huang
- 14:20–14:40 *An Empirical Study of Tokenization Strategies for Various Korean NLP Tasks*  
Kyubyong Park, Joohong Lee, Seongbo Jang and Dawoon Jung

**Saturday, Dec. 5 (all sessions are scheduled between 8:00 and 24:00 UTC +8) (continued)**

**14:00–15:00 Session 3B: NLP Applications I**

- 14:00–14:20 *BERT-Based Neural Collaborative Filtering and Fixed-Length Contiguous Tokens Explanation*  
Reinald Adrian Pugoy and Hung-Yu Kao
- 14:20–14:40 *Transformer-based Approach for Predicting Chemical Compound Structures*  
Yutaro Omote, Kyoumoto Matsushita, Tomoya Iwakura, Akihiro Tamura and Takashi Ninomiya
- 14:40–14:55 *Chinese Grammatical Correction Using BERT-based Pre-trained Model*  
Hongfei Wang, Michiki Kurosawa, Satoru Katsumata and Mamoru Komachi

**15:00–16:00 Session 4A: Information Extraction and Text Mining I**

- 15:00–15:20 *Neural Gibbs Sampling for Joint Event Argument Extraction*  
Xiaozhi Wang, Shengyu Jia, Xu Han, Zhiyuan Liu, Juanzi Li, Peng Li and Jie Zhou
- 15:20–15:40 *Named Entity Recognition in Multi-level Contexts*  
Yubo Chen, Chuhan Wu, Tao Qi, Zhigang Yuan and Yongfeng Huang

**15:00–16:00 Session 4B: Machine Translation and Multilinguality II**

- 15:00–15:20 *A General Framework for Adaptation of Neural Machine Translation to Simultaneous Translation*  
Yun Chen, Liangyou Li, Xin Jiang, Xiao Chen and Qun Liu
- 15:20–15:40 *UnihanLM: Coarse-to-Fine Chinese-Japanese Language Model Pretraining with the Unihan Database*  
Canwen Xu, Tao Ge, Chenliang Li and Furu Wei
- 15:40–16:00 *Towards a Better Understanding of Label Smoothing in Neural Machine Translation*  
Yingbo Gao, Weiyue Wang, Christian Herold, Zijian Yang and Hermann Ney

**Saturday, Dec. 5 (all sessions are scheduled between 8:00 and 24:00 UTC +8) (continued)**

**16:00–17:00 Session 5A: Semantics I & Resources and Evaluation I**

- 16:00–16:20 *Comparing Probabilistic, Distributional and Transformer-Based Models on Logical Metonymy Interpretation*  
Giulia Rambelli, Emmanuele Chersoni, Alessandro Lenci, Philippe Blache and Chu-Ren Huang
- 16:20–16:40 *AMR Quality Rating with a Lightweight CNN*  
Juri Opitz

**16:00–17:00 Session 5B: Summarization and Generation I**

- 16:00–16:20 *Generating Commonsense Explanation by Extracting Bridge Concepts from Reasoning Paths*  
Haozhe Ji, Pei Ke, Shaohan Huang, Furu Wei and Minlie Huang
- 16:20–16:40 *Unsupervised KB-to-Text Generation with Auxiliary Triple Extraction using Dual Learning*  
Zihao Fu, Bei Shi, Lidong Bing and Wai Lam

**17:00–18:00 Session 6A: Speech, Vision, Robotics, Multimodal Grounding I**

- 17:00–17:20 *Modality-Transferable Emotion Embeddings for Low-Resource Multimodal Emotion Recognition*  
Wenliang Dai, Zihan Liu, Tiezheng Yu and Pascale Fung
- 17:20–17:40 *All-in-One: A Deep Attentive Multi-task Learning Framework for Humour, Sarcasm, Offensive, Motivation, and Sentiment on Memes*  
Dushyant Singh Chauhan, Dhanush S R, Asif Ekbal and Pushpak Bhattacharyya

**Saturday, Dec. 5 (all sessions are scheduled between 8:00 and 24:00 UTC +8) (continued)**

**17:00–18:00 Session 6B: Summarization and Generation II**

- 17:00–17:20 *Identifying Implicit Quotes for Unsupervised Extractive Summarization of Conversations*  
Ryuji Kano, Yasuhide Miura, Tomoki Taniguchi and Tomoko Ohkuma
- 17:20–17:40 *Unsupervised Aspect-Level Sentiment Controllable Style Transfer*  
Mukuntha Narayanan Sundararaman, Zishan Ahmad, Asif Ekbal and Pushpak Bhattacharyya

**18:00–20:00 Dinner Break**

**20:00–21:00 Plenary Session: NLP Review 2020**

**21:00–22:00 Session 7A: Dialogue and Interactive Systems II**

- 21:00–21:20 *Energy-based Self-attentive Learning of Abstractive Communities for Spoken Language Understanding*  
Guokan Shang, Antoine Tixier, Michalis Vazirgiannis and Jean-Pierre Lorré
- 21:20–21:35 *Intent Detection with WikiHow*  
Li Zhang, Qing Lyu and Chris Callison-Burch

**22:00–23:00 Session 8A: NLP Applications II**

- 22:00–22:20 *A Systematic Characterization of Sampling Algorithms for Open-ended Language Generation*  
Moin Nadeem, Tianxing He, Kyunghyun Cho and James Glass
- 22:20–22:40 *Chinese Content Scoring: Open-Access Datasets and Features on Different Segmentation Levels*  
Yuning Ding, Andrea Horbach and Torsten Zesch
- 22:40–22:55 *Analysis of Hierarchical Multi-Content Text Classification Model on B-SHARP Dataset for Early Detection of Alzheimer's Disease*  
Renxuan Albert Li, Ihab Hajjar, Felicia Goldstein and Jinho D. Choi

**Saturday, Dec. 5 (all sessions are scheduled between 8:00 and 24:00 UTC +8) (continued)**

**22:00–23:00 Session 8B: Machine Translation and Multilinguality III**

- 22:00–22:20 *An Exploratory Study on Multilingual Quality Estimation*  
Shuo Sun, Marina Fomicheva, Frédéric Blain, Vishrav Chaudhary, Ahmed El-Kishky, Adithya Renduchintala, Francisco Guzmán and Lucia Specia
- 22:20–22:40 *English-to-Chinese Transliteration with Phonetic Auxiliary Task*  
Yuan He and Shay B. Cohen
- 22:40–22:55 *Predicting and Using Target Length in Neural Machine Translation*  
Zijian Yang, Yingbo Gao, Weiyue Wang and Hermann Ney

**22:00–23:00 Session 8C: Tagging, Chunking, Syntax, and Parsing II & Social Media I**

- 22:00–22:20 *Grounded PCFG Induction with Images*  
Lifeng Jin and William Schuler
- 22:20–22:40 *Heads-up! Unsupervised Constituency Parsing via Self-Attention Heads*  
Bowen Li, Taeuk Kim, Reinald Kim Amplayo and Frank Keller
- 22:40–23:00 *Building Location Embeddings from Physical Trajectories and Textual Representations*  
Laura Biester, Carmen Banea and Rada Mihalcea

**Saturday, Dec. 5 (all sessions are scheduled between 8:00 and 24:00 UTC +8) (continued)**

**23:00–24:00 Session 9A: Machine Learning for NLP II**

- 23:00–23:20 *Self-Supervised Learning for Pairwise Data Refinement*  
Gustavo Hernandez Abrego, Bowen Liang, Wei Wang, Zarana Parekh, Yinfei Yang and Yunhsuan Sung
- 23:20–23:40 *A Survey of the State of Explainable AI for Natural Language Processing*  
Marina Danilevsky, Kun Qian, Ranit Aharonov, Yannis Katsis, Ban Kawas and Prithviraj Sen
- 23:40–23:55 *Beyond Fine-tuning: Few-Sample Sentence Embedding Transfer*  
Siddhant Garg, Rohit Kumar Sharma and Yingyu Liang

**23:00–24:00 Session 9B: Speech, Vision, Robotics, Multimodal Grounding II**

- 23:00–23:20 *Multimodal Pretraining for Dense Video Captioning*  
Gabriel Huang, Bo Pang, Zhenhai Zhu, Clara Rivera and Radu Soricut
- 23:20–23:40 *Systematic Generalization on gSCAN with Language Conditioned Embedding*  
Tong Gao, Qi Huang and Raymond Mooney
- 23:40–24:00 *Are Scene Graphs Good Enough to Improve Image Captioning?*  
Victor Milewski, Marie-Francine Moens and Iacer Calixto

**Saturday, Dec. 5 (all sessions are scheduled between 8:00 and 24:00 UTC +8) (continued)**

**23:00–24:00 Session 9C: Summarization and Generation III**

- 23:00–23:20 *Systematically Exploring Redundancy Reduction in Summarizing Long Documents*  
Wen Xiao and Giuseppe Carenini
- 23:20–23:35 *A Cascade Approach to Neural Abstractive Summarization with Content Selection and Fusion*  
Logan Lebanoff, Franck Dernoncourt, Doo Soon Kim, Walter Chang and Fei Liu
- 23:35–23:50 *Mixed-Lingual Pre-training for Cross-lingual Summarization*  
Ruochen Xu, Chenguang Zhu, Yu Shi, Michael Zeng and Xuedong Huang

**23:00–24:00 Session 9D: Demo 1 (4 papers: 3 N.America, 1 Europe)**

**Sunday, Dec. 6 (all sessions are scheduled between 8:00 and 20:00 UTC +8)**

**8:00–9:00 Session 10A: Question Answering I**

- 8:00–8:20 *Point-of-Interest Oriented Question Answering with Joint Inference of Semantic Matching and Distance Correlation*  
Yifei Yuan, Jingbo Zhou and Wai Lam
- 8:20–8:35 *Leveraging Structured Metadata for Improving Question Answering on the Web*  
Xinya Du, Ahmed Hassan Awadallah, Adam Fourney, Robert Sim, Paul Bennett and Claire Cardie

**Sunday, Dec. 6 (all sessions are scheduled between 8:00 and 20:00 UTC +8) (continued)**

**8:00–9:00 Session 10B: Machine Translation and Multilinguality IV**

- 8:00–8:20 *English Intermediate-Task Training Improves Zero-Shot Cross-Lingual Transfer Too*  
Jason Phang, Iacer Calixto, Phu Mon Htut, Yada Pruksachatkun, Haokun Liu, Clara Vania, Katharina Kann and Samuel R. Bowman
- 8:20–8:35 *STIL - Simultaneous Slot Filling, Translation, Intent Classification, and Language Identification: Initial Results using mBART on MultiATIS++*  
Jack FitzGerald
- 8:35–8:50 *SimulMT to SimulST: Adapting Simultaneous Text Translation to End-to-End Simultaneous Speech Translation*  
Xutai Ma, Juan Pino and Philipp Koehn

**8:00–9:00 Session 10C: Summarization and Generation IV**

- 8:00–8:20 *Cue Me In: Content-Inducing Approaches to Interactive Story Generation*  
Faeze Brahman, Alexandru Petrusca and Snigdha Chaturvedi
- 8:20–8:40 *Liputan6: A Large-scale Indonesian Dataset for Text Summarization*  
Fajri Koto, Jey Han Lau and Timothy Baldwin
- 8:40–8:55 *Generating Sports News from Live Commentary: A Chinese Dataset for Sports Game Summarization*  
Kuan-Hao Huang, Chen Li and Kai-Wei Chang

**Sunday, Dec. 6 (all sessions are scheduled between 8:00 and 20:00 UTC +8) (continued)**

**8:00–9:00 Session 10D: Demo II (3 N.America, 1 Asia)**

**9:00–10:00 Session 11A: Information Retrieval and Document Analysis II**

9:00–9:20 *Massively Multilingual Document Alignment with Cross-lingual Sentence-Mover’s Distance*

Ahmed El-Kishky and Francisco Guzmán

9:20–9:40 *Improving Context Modeling in Neural Topic Segmentation*

Linzi Xing, Brad Hackinen, Giuseppe Carenini and Francesco Trebbi

9:40–9:55 *Contextualized End-to-End Neural Entity Linking*

Haotian Chen, Xi Li, Andrej Zukov Gregoric and Sahil Wadhwa

**9:00–10:00 Session 11B: Discourse and Pragmatics I**

9:00–9:20 *DAPPER: Learning Domain-Adapted Persona Representation Using Pretrained BERT and External Memory*

Prashanth Vijayaraghavan, Eric Chu and Deb Roy

9:20–9:40 *Event Coreference Resolution with Non-Local Information*

Jing Lu and Vincent Ng

9:40–9:55 *Neural RST-based Evaluation of Discourse Coherence*

Grigorii Guz, Peyman Bateni, Darius Muglich and Giuseppe Carenini

**Sunday, Dec. 6 (all sessions are scheduled between 8:00 and 20:00 UTC +8) (continued)**

**9:00–10:00 Session 11C: Resources and Evaluation II**

- 9:00–9:20 *Asking Crowdworkers to Write Entailment Examples: The Best of Bad Options*  
Clara Vania, Ruijie Chen and Samuel R. Bowman

**10:00–11:00 Session 12A: Question Answering II**

- 10:00–10:20 *MaP: A Matrix-based Prediction Approach to Improve Span Extraction in Machine Reading Comprehension*  
Huashao Luo, Yu Shi, Ming Gong, Linjun Shou and Tianrui Li
- 10:20–10:40 *Answering Product-related Questions with Heterogeneous Information*  
Wenxuan Zhang, Qian Yu and Wai Lam

**11:00–12:00 Keynote**

**12:00–14:00 Lunch Break**

**14:00–15:00 Session 13A: Semantics II**

- 14:00–14:20 *Two-Step Classification using Recasted Data for Low Resource Settings*  
Shagun Uppal, Vivek Gupta, Avinash Swaminathan, Haimin Zhang, Debanjan Mahaata, Rakesh Gosangi, Rajiv Ratn Shah and Amanda Stent
- 14:20–14:35 *Explaining Word Embeddings via Disentangled Representation*  
Keng-Te Liao, Cheng-Syuan Lee, Zhong-Yu Huang and Shou-de Lin

**Sunday, Dec. 6 (all sessions are scheduled between 8:00 and 20:00 UTC +8) (continued)**

**14:00–15:00 Session 13B: Knowledge Graph II**

- 14:00–14:20 *Multi-view Classification Model for Knowledge Graph Completion*  
Wenbin Jiang, Mengfei Guo, Yufeng Chen, Ying Li, Jinan Xu, Yajuan Lyu and Yong Zhu
- 14:20–14:40 *Knowledge-Enhanced Named Entity Disambiguation for Short Text*  
Zhifan Feng, Qi Wang, Wenbin Jiang, Yajuan Lyu and Yong Zhu

**15:00–16:00 Session 14A: Information Extraction and Text Mining II**

- 15:00–15:20 *More Data, More Relations, More Context and More Openness: A Review and Outlook for Relation Extraction*  
Xu Han, Tianyu Gao, Yankai Lin, Hao Peng, Yaoliang Yang, Chaojun Xiao, Zhiyuan Liu, Peng Li, Jie Zhou and Maosong Sun
- 15:20–15:40 *Robustness and Reliability of Gender Bias Assessment in Word Embeddings: The Role of Base Pairs*  
Haiyang Zhang, Alison Sneyd and Mark Stevenson

**15:00–16:00 Session 14B: NLP Applications III**

- 15:00–15:20 *ExpanRL: Hierarchical Reinforcement Learning for Course Concept Expansion in MOOCs*  
Jifan Yu, Chenyu Wang, Gan Luo, Lei Hou, Juanzi Li, Jie Tang, Minlie Huang and Zhiyuan Liu
- 15:20–15:35 *Vocabulary Matters: A Simple yet Effective Approach to Paragraph-level Question Generation*  
Vishwajeet Kumar, Manish Joshi, Ganesh Ramakrishnan and Yuan-Fang Li

**Sunday, Dec. 6 (all sessions are scheduled between 8:00 and 20:00 UTC +8) (continued)**

**15:00–16:00 Session 14C: Social Media and Computational Social Science II**

15:00–15:20 *From Hero to Zéroe: A Benchmark of Low-Level Adversarial Attacks*

Steffen Eger and Yannik Benz

15:20–15:35 *Point-of-Interest Type Inference from Social Media Text*

Danae Sánchez Villegas, Daniel Preotiuc-Pietro and Nikolaos Aletras

**16:00–17:00 Session 15A: Information Extraction and Text Mining III**

16:00–16:20 *Reconstructing Event Regions for Event Extraction via Graph Attention Networks*

Pei Chen, Hang Yang, Kang Liu, Ruihong Huang, Yubo Chen, Taifeng Wang and Jun Zhao

16:20–16:35 *Recipe Instruction Semantics Corpus (RISeC): Resolving Semantic Structure and Zero Anaphora in Recipes*

Yiwei Jiang, Klim Zaporojets, Johannes Deleu, Thomas Demeester and Chris Develder

**16:00–17:00 Session 15B: NLP Applications IV**

16:20–16:35 *Stronger Baselines for Grammatical Error Correction Using a Pretrained Encoder-Decoder Model*

Satoru Katsumata and Mamoru Komachi

**17:00–18:00 Session 16A: Resources and Evaluation III**

17:00–17:20 *Sina Mandarin Alphabetical Words: A Web-driven Code-mixing Lexical Resource*

Rong Xiang, Mingyu Wan, Qi Su, Chu-Ren Huang and Qin Lu

17:20–17:40 *IndoNLU: Benchmark and Resources for Evaluating Indonesian Natural Language Understanding*

Bryan Wilie, Karissa Vincentio, Genta Winata, Samuel Cahyawijaya, Xiaohong Li, Zhi Yuan Lim, Sidik Soleman, Rahmad Mahendra, Pascale Fung, Syafri Bahar and Ayu Purwarianti

17:40–18:00 *Happy Are Those Who Grade without Seeing: A Multi-Task Learning Approach to Grade Essays Using Gaze Behaviour*

Sandeep Mathias, Rudra Murthy, Diptesh Kanodia, Abhijit Mishra and Pushpak Bhattacharyya

**Sunday, Dec. 6 (all sessions are scheduled between 8:00 and 20:00 UTC +8) (continued)**

**17:00–18:00 Session 16B: Machine Learning for NLP III**

- 17:00–17:20 *Multi-Source Attention for Unsupervised Domain Adaptation*  
Xia Cui and Danushka Bollegala
- 17:20–17:35 *Compressing Pre-trained Language Models by Matrix Decomposition*  
Matan Ben Noach and Yoav Goldberg

**18:00–19:00 Session 17A: NLP Applications V & Question Answering III**

- 18:00–18:20 *You May Like This Hotel Because ...: Identifying Evidence for Explainable Recommendations*  
Shin Kanouchi, Masato Neishi, Yuta Hayashibe, Hiroki Ouchi and Naoaki Okazaki
- 18:20–18:40 *A Unified Framework for Multilingual and Code-Mixed Visual Question Answering*  
Deepak Gupta, Pabitra Lenka, Asif Ekbal and Pushpak Bhattacharyya

**18:00–19:00 Session 17B: Social Media and Computational Social Science III**

- 18:00–18:20 *Toxic Language Detection in Social Media for Brazilian Portuguese: New Dataset and Multilingual Analysis*  
João Augusto Leite, Diego Silva, Kalina Bontcheva and Carolina Scarton
- 18:20–18:35 *Measuring What Counts: The Case of Rumour Stance Classification*  
Carolina Scarton, Diego Silva and Kalina Bontcheva

**19:00–20:00 Closing (Best Paper, Future conf.)**



# Touch Editing: A Flexible One-Time Interaction Approach for Translation

**Qian Wang<sup>1,2</sup>, Jiajun Zhang<sup>1,2</sup>, Lemao Liu<sup>3</sup>, Guoping Huang<sup>3</sup>, Chengqing Zong<sup>1,2</sup>**

<sup>1</sup>National Laboratory of Pattern Recognition, CASIA, Beijing, China

<sup>2</sup>University of Chinese Academy of Sciences, Beijing, China

<sup>3</sup>Tencent AI Lab, Shenzhen, China

{qian.wang, jjzhang, cqzong}@nlpr.ia.ac.cn, {redmondliu, donkeyhuang}@tencent.com

## Abstract

We propose a touch-based editing method for translation, which is more flexible than traditional keyboard-mouse-based translation post-editing. This approach relies on touch actions that users perform to indicate translation errors. We present a dual-encoder model to handle the actions and generate refined translations. To mimic the user feedback, we adopt the TER algorithm comparing between draft translations and references to automatically extract the simulated actions for training data construction. Experiments on translation datasets with simulated editing actions show that our method significantly improves original translation of Transformer (up to 25.31 BLEU) and outperforms existing interactive translation methods (up to 16.64 BLEU). We also conduct experiments on post-editing dataset to further prove the robustness and effectiveness of our method.

## 1 Introduction

Neural machine translation (NMT) has made great success during the past few years (Sutskever et al., 2014; Bahdanau et al., 2014; Wu et al., 2016; Vaswani et al., 2017), but automatic machine translation is still far from perfect and cannot meet the strict requirements of users in real applications (Petrushkov et al., 2018). Many notable human-machine interaction approaches have been proposed for allowing professional translators to improve machine translation results (Wuebker et al., 2016; Knowles and Koehn, 2016; Hokamp and Liu, 2017). As an instance of such approaches, post-editing directly requires translators to modify outputs from machine translation (Simard et al., 2007). However, traditional post-editing requires intensive keyboard interaction, which is inconvenient on mobile devices.

Grangier and Auli (2018) suggest a one-time interaction approach with lightweight editing ef-

forts, QuickEdit, in which users are asked to simply mark incorrect words in a translation hypothesis for one time in the hope that the system will change them. QuickEdit delivers appealing improvements on draft hypotheses while maintaining the flexibility of human-machine interaction. Unfortunately, only marking incorrect words is far from adequate: for example, it does not indicate the missing information beyond the original hypothesis, which is a typical issue called under-translation in machine translation (Tu et al., 2016).

In this paper, we propose a novel one-time interaction method called Touch Editing, which is flexible for users and more adequate for a system to generate better translations. Inspired by human editing process, the proposed method relies on a series of touch-based actions including SUBSTITUTION, DELETION, INSERTION and REORDERING. These actions do not include lexical information and thus can be flexibly provided by users through various of gestures on touch screen devices. By using these actions, our method is able to capture the editing intention from users to generate better translations: for instance, INSERTION indicates a word is missing at a particular position, and our method is expected to insert the correct word. To this end, we present a neural network model by augmenting Transformer (Vaswani et al., 2017) with an extra encoder for a hypothesis and its actions. Since it is impractical to manually annotate large-scale action dataset to train the model, we thereby adopt the algorithm of TER (Snover et al., 2006) to automatically extract actions from a draft hypothesis and its reference.

To evaluate our method, we conduct simulated experiments on translation datasets the same as in other works (Denkowski et al., 2014; Grangier and Auli, 2018). The results demonstrate that our method can address the well-known challenging issues in machine translation including over-

Source x	weite wege müsse proctor für die nahrungsmittelbeschaffung nicht gehen .
Reference y	proctor does not have to travel far to buy food .
QuickEdit	Hypothesis $y'$ travel far does not necessary to proctor <del>for</del> food <del>supply</del> . Result travel far does not require to proctor food supplies .
Touch Editing	Hypothesis $y'$ travel far does not necessary to proctor <del>for</del> food <del>supply</del> . Modified $m(y')$ proctor does not necessary to travel far for <INS> food supply . Action Sequence a - - - S - - - S I - D - Result proctor does not have to travel far to buy food .

Figure 1: Example of interaction methods. QuickEdit allows users to mark incorrect words. Our method introduces more flexible actions.  $m(y')$  is modified from  $y'$  by applying reordering actions and inserting a special token  $\langle\text{INS}\rangle$  to keep alignment with the action sequence  $a$  which contains actions like **SUBSTITUTION**, **INSERTION** and **DELETION**. “-” denotes the word in that position is unmarked. Our method then generates a refined translation based on the modified hypothesis  $m(y')$  and the action sequence  $a$ .

translation, under-translation and mis-ordering, and thus it outperforms Transformer and QuickEdit by a margin up to 25.31 and 16.64 BLEU points respectively. In addition, experiments on post-editing dataset further prove the effectiveness and robustness of our method. Finally, we implement a real application on mobile phones to discuss the usability in real scenarios.

## 2 Touch Editing Approach

### 2.1 Actions

QuickEdit allows translators to mark incorrect words which they expect the system to change (Grangier and Auli, 2018). However, as shown in Figure 1, the information is inadequate for a system to correct a translation hypothesis, especially when it comes to under-translation, in which the system is hardly to predict missing words into hypotheses.

To achieve better adequacy, we take human editing habits into consideration. As shown in Figure 1, a human translator may insert, delete, substitute or reorder some words to correct errors of under-translation, over-translation, mis-translation and mis-ordering in an original translation hypothesis. Based on human editing process, we define a set of actions to represent human editing intentions:

- **INSERTION**: a new word should be inserted into a given position.
- **DELETION**: a word at a specific position should be deleted.

- **SUBSTITUTION**: a word should be substituted by another word.
- **REORDERING**: a segment of words should be moved to another position.

In Touch Editing, these actions can be performed by human translators on a given machine hypothesis to indicate translation errors. To keep the flexibility of interactions, for SUBSTITUTION and INSERTION actions, our method allows users to only indicate which word should be substitute or in which position a word should be inserted. The light-weight interaction in Touch Editing is non-lexical, i.e., it does not require any keyboard inputs, and thus can be adopted to mobile devices with touch screens.

### 2.2 Model

Our model seeks to correct translation errors of an original hypothesis  $y'$  based on actions  $\mathcal{A}$  provided by human translator.

To make full use of the actions, we firstly modify the original hypothesis by applying  $\mathcal{A}$  on  $y'$  to obtain  $\mathcal{A}(y')$ :

$$\mathcal{A}(y') = \langle m(y'), \mathbf{a} \rangle. \quad (1)$$

Specifically, as shown in Figure 1,  $m(y')$  is modified from  $y'$  by reordering the segment in gray color and inserting a token  $\langle\text{INS}\rangle$ , and thus the **REORDERING** actions is implicitly included in

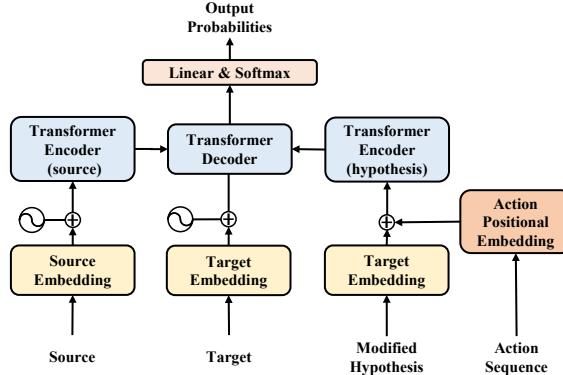


Figure 2: Model architecture. We add a *hypothesis encoder* (the right part) into Transformer which differs from *source encoder* (the left part) in positional embedding. We use learned action positional embedding instead of the sinusoids.

$m(\mathbf{y}')$ . The action sequence  $\mathbf{a}$  below  $m(\mathbf{y}')$  contains **SUBSTITUTION**, **INSERTION** and **DELETION** at the corresponding position.

We then use a neural network model to generate a translation  $\mathbf{y}$  for the source sentence  $\mathbf{x}$ , the hypothesis  $\mathbf{y}'$  and the actions  $\mathcal{A}$ :

$$P(\mathbf{y} | \mathbf{x}, \mathbf{y}', \mathcal{A}; \theta) = \prod_{n=1}^N P(y_n | y_{<n}, \mathbf{x}, m(\mathbf{y}'), \mathbf{a}; \theta). \quad (2)$$

As shown in Figure 2, the neural network model we developed is a dual encoder model based on Transformer similar to Tebbifakhr et al. (2018). Specifically, besides encoding the source sentence  $\mathbf{x}$  with *source encoder* (the left part of Figure 2), our model additionally encodes  $\mathcal{A}(\mathbf{y}')$  with an extra *hypothesis encoder* (the right part of Figure 2) and integrates the encoded representations into decoding network using dual multi-head attention.

**Encoding  $\mathcal{A}(\mathbf{y}')$**  As shown in the right part of Figure 2, the *hypothesis encoder* firstly embeds  $m(\mathbf{y}')$  with length  $l$  in distributed space using the same word embedding as in *decoder*, which is denoted as  $\mathbf{w} = \{w_1, \dots, w_l\}$ . Then it encodes  $\mathbf{a} = \{a_1, \dots, a_l\}$  with learned positional embedding according to the specific actions. As shown in Figure 3, the action positional embedding includes four embedding matrixes corresponding to three action types and a none action for positions without any action. For the  $i$ th position of  $\mathbf{a}$ , the encoder chooses an embedding matrix based on the action type of  $a_i$  and selects the  $i$ th row of the matrix as the positional embedding vector, which is denoted as  $p_i$ :

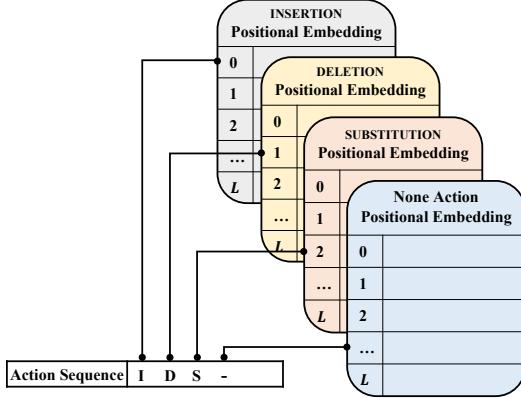


Figure 3: Action positional embedding. The model firstly chooses an embedding matrix according to the action at position  $i$ , then lookups the  $i$ th row of the matrix as the positional embedding of position  $i$ .  $L$  is the maximum length of sentences.

$$p_i = \begin{cases} PE_{\text{INSERTION}}(i) & \text{if } a_i = \text{I} \\ PE_{\text{DELETION}}(i) & \text{if } a_i = \text{D} \\ PE_{\text{SUBSTITUTION}}(i) & \text{if } a_i = \text{S} \\ PE_{\text{None}}(i) & \text{if } a_i = \text{-} \end{cases} \quad (3)$$

Where  $PE_*$  denote the action positional embedding matrixes in Figure 3. The learned action positional embedding is used in *hypothesis encoder* to replace the fixed sinusoids positional encoding in Transformer encoder. Next, the encoder adds the word embedding  $\mathbf{w}$  and the action positional embedding  $\mathbf{p}$  to obtain input embedding  $\mathbf{e} = \{w_1 + p_1, \dots, w_l + p_l\}$ . The following part of *hypothesis encoder* lies the same as Transformer encoder.

**Decoding** The output of *hypothesis encoder*, together with the output of *source encoder*, are fed into the *decoder*. To combine both of the encoders' outputs, we apply dual multi-head attention in each layer of decoder: the attention sub-layer attends to both encoders' outputs by performing multi-head attention respectively:

$$\begin{aligned} A_{src} &= \text{MultiHead}(Q_{tgt}, K_{src}, V_{src}) \\ A_{hyp} &= \text{MultiHead}(Q_{tgt}, K_{hyp}, V_{hyp}) \end{aligned} \quad (4)$$

Where  $Q_{tgt}$  is coming from previous layer of the decoder,  $K_{src}$  and  $V_{src}$  matrixes are final representations of the *source encoder* while  $K_{hyp}$  and  $V_{hyp}$  matrixes are final representations of the *hypothesis encoder*. The two attention vectors  $A_{src}$  and  $A_{hyp}$  are then averaged to replace encoder-decoder attention in Transformer, resulting in the input of next layer.

**Training** The overall model, which includes a *source encoder*, a *hypothesis encoder* with action positional embedding, and a *decoder*, is jointly trained. We maximize the log-likelihood of the reference sentence  $y$  given the source sentence  $x$ , the initial hypothesis  $y'$ , and the corresponding actions  $\mathcal{A}$ . By applying  $\mathcal{A}$  on  $y'$ , the training objective becomes:

$$\hat{\theta} = \arg \max_{\theta} \left\{ \sum_{\mathcal{D}} \log P(y | x, m(y'), a; \theta) \right\}. \quad (5)$$

where  $\mathcal{D}$  is the training dataset consists of quadruplets like (source  $x$ , modified hypothesis  $m(y')$ , action sequence  $a$ , target  $y$ ). We use Adam optimizer (Kingma and Ba, 2014), an extension of stochastic gradient descent (Bottou, 1991), to train the model.

After training, the model with parameter  $\hat{\theta}$  is then used in inference phase to generate refined translations for test data, which consists of triplets like (source  $x$ , modified hypothesis  $m(y')$ , action sequence  $a$ ).

### 3 Automatic Data Annotation

The actions we defined in Section 2.1 can be provided by human translators in real applications. However, it is impractical to manually collect a large scale annotated dataset for training our model. Thus we resort to propose an approach to automatically extract editing actions from a machine translation hypothesis and its corresponding reference.

To make our method powerful, the number of editing actions which convert a hypothesis to its

---

#### Algorithm 1 Extracting actions with TER

---

**Input:** hypothesis  $y'$ , reference  $y$   
 $m(y') \leftarrow y'$   
 $a \leftarrow$  Empty action sequence  
**repeat**  
    Find reordering  $r$  that most reduces min-edit-distance( $m(y')$ ,  $y$ )  
    **if**  $r$  reduces edit distance **then**  
         $m(y') \leftarrow$  applying  $r$  to  $m(y')$   
    **end if**  
**until** no beneficial reordering remains  
 $a \leftarrow \text{min-edit}(m(y'), y)$   
 $m(y') \leftarrow$  insert  $\langle \text{INS} \rangle$  into  $m(y')$  based on  $a$   
**Output:**  $m(y')$ ,  $a$

---

reference is minimal as presented in Section 2.1. Snover et al. (2006) study this problem and point out that its optimal solution is NP-hard (Lopresti and Tomkins, 1997; Shapira and Storer, 2002). To optimize the number of editing actions, they instead propose an approximate algorithm based on minimal edit distance. The basic idea of their algorithm can be explained as follows. It repeatedly modifies the intermediate string by applying reordering actions which is greedily found to mostly reduce the edit distance between the intermediate string and the reference, until no more beneficial reordering remains.

In this paper, we adopt the basic idea of Snover et al. (2006) to automatically extract actions. As shown in Algorithm 1, given a reference and a hypothesis, the algorithm repeatedly reorders words to reduce the word-level minimal edit distance between reference  $y$  and modified hypothesis  $m(y')$  until no beneficial reordering remains. With the modified hypothesis  $m(y')$ , the algorithm then calculates the editing action sequence  $a$  that minimize the word-level edit distance between  $m(y')$  and  $y$  (see Action Sequence  $a$  in Figure 1). It finally inserts special token  $\langle \text{INS} \rangle$  to keep alignment between the modified hypothesis and the action sequence (see Modified  $m(y')$  in Figure 1). The output of the algorithm, which is a tuple of modified hypothesis and action sequence, together with the source sentence and its reference, are used to train our model as described in Section 2.2.

### 4 Experiment

We conduct simulated experiment on translation datasets. Specifically, we translate the source sentences in translation datasets with a pre-trained Transformer model and build the training data with simulated human feedback using algorithm described in Section 3.

#### 4.1 Dataset and Settings

The experiment is conducted on three translation datasets: the IWSLT'14 English-German dataset (Cettolo et al., 2014), the WMT'14 English-German dataset (Bojar et al., 2014) and the WMT'17 Chinese-English dataset (Ondrej et al., 2017). The IWSLT'14 English-German dataset consists of 170k sentence pairs from TED talk subtitles. We use *dev2010* as validation set which contains 887 sentent pairs, and a concatenation of *tst2010*, *tst2011* and *tst2012* as test set which con-

Model	IWSLT'14				WMT'14				WMT'17			
	EN-DE		DE-EN		EN-DE		DE-EN		EN-ZH		ZH-EN	
	BLEU	TER										
ConvS2S <sup>†</sup>	24.20	-	27.40	-	25.20	-	29.70	-	-	-	-	-
QuickEdit <sup>†</sup>	30.80	-	34.60	-	36.60	-	41.30	-	-	-	-	-
Transformer	27.40	0.52	33.17	0.45	26.69	0.56	31.73	0.48	32.53	0.55	21.89	0.61
QuickEdit <sup>‡</sup>	34.33	0.43	40.13	0.39	37.00	0.43	41.48	0.39	41.20	0.43	29.78	0.51
Touch Baseline	34.48	0.42	40.09	0.35	33.92	0.43	39.47	0.37	38.96	0.42	29.17	0.51
Touch Editing	<b>44.25</b>	<b>0.32</b>	<b>50.39</b>	<b>0.29</b>	<b>50.49</b>	<b>0.28</b>	<b>56.47</b>	<b>0.24</b>	<b>57.84</b>	<b>0.28</b>	<b>45.67</b>	<b>0.33</b>

Table 1: Results of different systems measured in BLEU and TER. <sup>†</sup> denotes the results from Quick Edit. QuickEdit<sup>‡</sup> is our reimplementation based on Transformer. Touch baseline is the result modified from initial hypothesis by deleting and reordering words. Touch Editing is our model trained with all actions described in Section 2.1.

tains 4698 sentence pairs. For WMT’14 English-German dataset, we use the same data and pre-processing as (Luong et al., 2015). The dataset consists of 4.5M sentence pairs for training<sup>1</sup>. We take *newstest2013* for validation and *newstest2014* for testing. For Chinese to English dataset, we use CWMT portion which is a subset of WMT’17 training data containing 9M sentence pairs. We validate on *newsdev2017* and test on *newstest2017*.

As for vocabulary, the English and German datasets are encoded using byte-pair encoding (Sennrich et al., 2015) with a shared vocabulary of 8k tokens for IWSLT’14 and 32k tokens for WMT’14. For Chinese to English dataset, the English vocabulary is set to 30k subwords, while the Chinese data is tokenized into character level and the vocabulary is set to 10k characters. Note that even with subword units or character units, the actions are marked in word level, i.e. all units from a given word share the same actions.

We train the models with two settings. For the larger WMT English-German and English-Chinese dataset, we borrow the Transformer base parameter set of Vaswani et al. (2017), which contains 6 layers for encoders and decoder respectively. The multi-head attention of each layer contains 8 heads. The word embedding size is set to 512 and the feed-forward layer dimension is 2048. For the smaller IWSLT dataset, we use 3 layers for each component and multi-head attention with 4 heads in each layer. The word embedding size is 256 and the feedforward layers’ hidden size is 1024. We also apply label smoothing  $\sigma_{ls} = 0.1$  and dropout  $p_{dropout} = 0.1$  during training. All models are

trained from scratch with corresponding training data, e.g., parallel data for Transformer baseline model and annotated data for Touch Editing.

## 4.2 Main Results

We report the results of different systems including Transformer and QuickEdit. The Transformer model is tested on bitext data, i.e., the model directly generates translations based on source sentences. As for the QuickEdit, we followed the settings of Grangier and Auli (2018), in which they mark all words in initial translation results that do not appear in the references as incorrect, and use the QuickEdit model to generate refined translations. In Touch Baseline setting, we use the algorithm described in Section 3 to obtain the actions respect to initial translations and references, and then apply reordering and deletion actions to obtain refined translations. The Touch Edit setting accesses the same information as Touch Baseline but uses the neural model described in Section 2.2 to handle the actions. Note that the original QuickEdit model is based on ConvS2S, and thus we reimplement it based on Transformer to keep the *fairness* of comparison<sup>2</sup>.

As shown in Table 1, our model strongly outperforms other systems. As for BLEU score, our model achieves up to +25.31 than Transformer and +16.64 than QuickEdit. Our model also significantly reduces TER by -0.28 and -0.18 comparing to Transformer and QuickEdit.

We also notice that the improvement on the smaller IWSLT’14 dataset (up to 17.22) is not as

<sup>1</sup>We use the pre-processed data from <https://nlp.stanford.edu/projects/nmt/>

<sup>2</sup>In fact, the comparison is still unfair because QuickEdit and our method access more supervised information than Transformer from simulated human feedback, which is the nature of interaction settings.

	Reordering	RIBES
Transformer	4672	79.97
QuickEdit	4799	84.33
Touch Editing	<b>650</b>	<b>90.50</b>

Table 2: Word reordering quality, measured in number of word reorderings required to align to references, and RIBES score.

significant as that on the larger WMT’14 dataset (up to 24.74) and WMT’17 dataset (up to 25.31). This observation is in consistent with QuickEdit, which also gains lower improvement on the smaller dataset. The reason, as described in Grangier and Auli (2018), is that the underlying machine translation model is overfitted on the smaller 170k dataset. Thus the translation output requires less edits on which we build simulated editing action dataset. The limited supervised data further impacts the model quality and final results.

### 4.3 Analysis

To further investigate the model capacity, we conduct four experiments on WMT’14 English to German dataset. We analyze the factors that bring the remarkable improvement by modeling coverage, reordering quality and accuracy of each action type. We also test our model with limited number of actions to evaluate the model usability with partial feedback.

**Reordering** We evaluate the word reordering quality of our model, compared with Transformer and QuickEdit. We adopt two automatic evaluation metrics. One metric is based on monolingual alignment. We firstly align model hypotheses and references with TER, and then count the number of words that should be reordered. As shown by Reordering in Table 2, the output of our model requires less word reorderings to align with reference.

The other metric is RIBES (Isozaki et al., 2010), which is based on rank correlation. As shown in Table 2, our method outperforms the other two systems with 90.50 versus 79.97 for Transformer and 84.33 for QuickEdit.

**Accuracy** As described in Section 2.1, the actions of our method represent human editing intentions, i.e., they indicate errors in original hypothesis and our model is expected to correct these errors based on editing actions. To evaluate the accuracy

		Total	Correct	Accuracy
Quick Edit	Deletion	6438	4440	68.97%
	Insertion	4430	681	15.37%
	Substitution	20858	5030	24.12%
Touch Editing	Deletion	6438	6383	99.15%
	Insertion	4430	1609	36.32%
	Substitution	20858	6645	31.86%

Table 3: Accuracy of actions. Total means number of actions to transform the draft machine translations into references. Correct means how many words are corrected (or deleted) by the model.

of INSERTION, DELETION and SUBSTITUTION, we first use TER to align machine translation hypotheses and references, as well as our model’s outputs and references. With the references as intermediates, we then align our model’s outputs and original machine translations. With the alignment result, we directly check whether the words with actions are corrected or not to calculate the accuracy of the three actions. To make a complete comparison, we also analyze the results of QuickEdit and calculate the accuracy.

As shown in Table 3, our model achieves the accuracy of 99.15% for deletion<sup>3</sup>, 36.32% for insertion and 31.86% for substitution. The high deletion accuracy shows that our model indeed learns to delete over-translated words. For insertion and substitution, the actions only indicate where to insert or substitute, and do not provide any ground truth. Since the self-attention mechanism in Transformer is good at word sense disambiguation (Tang et al., 2018a,b), our model is able to select correct words to insert or substitute.

**Partial Feedback** The model we train and test is based on all actions, i.e., all translation errors of the initial hypotheses are marked out. However, a human translator may not provide all marks. In fact, the feedback of human translators is hard to predict, and vary with different translators.

In this case, we test our model with simulated partial feedback. We train our model with all actions and randomly select 0%, 5%, . . . 100% of actions in test set to simulate human behavior. To further investigate the effect of partial feedback

<sup>3</sup>We do not explicitly remove words that marked as DELETION and the neural model is responsible for making final decision whether these words should be deleted. It might slightly hurt BLEU and accuracy but potentially generates more fluent translations.

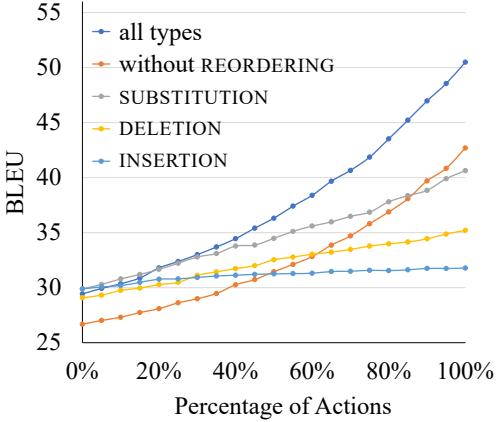


Figure 4: Results of partial feedback measured in BLEU score. We train five models to investigate the effects of partial feedback on different actions.

on different actions, we train three extra models with specific kinds of actions: INSERT, DELETE and SUBSTITUTE. We then randomly select part of each kind of actions to test the model. Note that the REORDERING actions are always enabled since they are operated on a segment of words and cannot be partially disabled. To investigate the effect of REORDERING actions, we also train a model without reordering and partially select three kinds of actions to test the model.

As shown in Figure 4, for the model trained with all actions, the BLEU scores increases from 29.43 (with reordering only) to 50.49 (with all actions) as more actions are provided. For the models trained with specific kinds of actions and the model trained without reordering, the observation is similar.

#### 4.4 Experiments on Post-Editing Data

In previous sections, our model is tested and analyzed on automatic machine translation datasets. However, in post-editing scenarios, our model faces three major challenges: action inconsistency, data inconsistency and model inconsistency. For action inconsistency, the editing actions to train our model are extracted from machine predictions and references. The references in our training data are written by human from scratch, while in post-editing the references (human post-edited results) are revisions of machine translations, and thus the editing actions might be different. For data inconsistency, our model is trained on dataset of News domain (WMT) or TED talks (IWSLT). However in real world, data may be from any other domains. For model inconsistency, we use Transformer to build our training data while the translation model used

	WMT 16		WMT 17	
	BLEU	TER	BLEU	TER
MT	62.48	0.24	62.83	0.24
QuickEdit	67.14	0.19	69.22	0.18
Touch Editing	<b>82.05</b>	<b>0.09</b>	<b>82.88</b>	<b>0.09</b>

Table 4: Results on post-editing dataset in terms of BLEU and TER.

in real applications may be different.

To investigate the performance facing the three challenges, we test our model on WMT English-German Automatic Post-Editing (APE) dataset in IT domain using data from WMT’16 (Bojar et al., 2016) and WMT’17 (Ondrej et al., 2017). The test data consists of triplets like (*source, machine translation, human post-edit*), in which the machine translation is generated with a PBSMT system. We use the algorithm of Section 3 to extract actions from machine translations and human post-edited sentences. With the actions and original machine translations, we use the model trained on WMT’14 English-German dataset in Section 4 to generate refined translations. To make a comparison, we also evaluate QuickEdit with the same setting.

Table 4 summarizes the results on post-editing dataset. It is clear to see that even with the three kinds of inconsistency, our model still gains significant improvements of up to 20.05 BLEU than the raw machine translation system (PBSMT). As for QuickEdit, the improvement on post-editing dataset (about 4-7 BLEU) is smaller than that on translation dataset (about 11 BLEU). We conjecture that the stable improvement of our method is due to more flexible action types. With the detailed editing actions, the model is competent to correct various of errors in draft machine translations, and thus leads to the robustness and effectiveness of our method.

#### 4.5 Discussion on Real Scenarios

So far, the experiments we conducted are based on simulated human feedbacks, in which the actions are extracted from initial machine translation results and their corresponding references to simulate human editing actions. Thus in our simulated setting, the references are used in inference phase to simulate human behavior, as in other interaction methods (Denkowski et al., 2014; Marie and Max, 2015; Grangier and Auli, 2018). These experiments show that our method can significantly

improve the initial translation with simulated actions. However, whether the actions are convenient to perform is a key point in real applications.

To investigate the usability and applicable scenarios of our method, we implement a real mobile application on iPhone, in which the actions can be performed on multi-touch screens. For a given source sentence, the application provides an initial machine translation. The text area of translation can respond to several gestures<sup>4</sup>: **Tap** indicated a missing word should be inserted into the nearest space between two words; **Swipe** on a word indicated that the word should be deleted; **Long-Press** a word means the word should be substituted with other word; **Pan** can drag a word to another position.

We conduct a free-use study with four participants, in which the participants are asked to translate 20 sentences randomly selected from LDC Chinese-English test set with (1) Touch Editing or (2) keyboard input after 5 minutes to get familiar with the application. We observe that the users with Touch Editing tends to correct an error for multiple times when the system cannot predict a word they want, while the users with keyboard input tends to modify more content of initial translation and spend more time on choosing words. We then conduct an unstructured interview on the usability of our method. The result of the interview shows that Touch Editing is convenient and intuitive but lack of ability of generating final accurate translation. It can be treated as a light-weight proofreading method, and suitable for Pre-Post-Editing (Marie and Max, 2015).

## 5 Related Work

Post-editing is a pragmatic method that allows human translators to directly correct errors in draft machine translations (Simard et al., 2007). Comparing to purely manual translation, it achieves higher productivity while maintaining the human translation quality (Plitt and Masselot, 2010; Federico et al., 2012).

Many notable works introduce different levels of human-machine interactions in post-editing. Barachina et al. (2009) propose a prefix-based interactive method which enable users to correct the first translation error from left to right in each iteration.

<sup>4</sup>These gestures are explicit and directly supported by Apple iOS devices: <https://developer.apple.com/documentation/uikit/uigesturerecognizer>

Green et al. (2014) implement a prefix-based interactive translation system and Huang et al. (2015) adopt the prefix constrained translation candidates into a novel input method for translators. Peris et al. (2017) further extend this idea to neural machine translation.

The prefix-based protocol is inflexible since users have to follow the left-to-right order. To overcome the weakness of prefix-based approach, González-Rubio et al. (2016); Cheng et al. (2016) introduce interaction methods that allow users to correct errors at arbitrary position in a machine hypothesis, while Weng et al. (2019) also preventing repeat mistakes by memorizing revision actions. Hokamp and Liu (2017) propose grid beam search to incorporate lexical constraints like words and phrases provided by human translators and force the constraints to appear in hypothesis.

Recently, some researchers resort to more flexible interactions, which only require mouse click or touch actions. For example, Marie and Max (2015); Domingo et al. (2016) propose interactive translation methods which ask user to select correct or incorrect segments of a translation with mouse only. Similar to our work, Grangier and Auli (2018) propose a mouse based interactive method which allows users to simply mark the incorrect words in draft machine hypotheses and expect the system to generate refined translations. Herbig et al. (2019, 2020) propose a multi-modal interface for post-editors which takes pen, touch, and speech modalities into consideration.

The protocol that given an initial translation to generate a refined translation, is also used in polishing mechanism in machine translation (Xia et al., 2017; Geng et al., 2018) and automatic post-editing (APE) task (Lagarda et al., 2009; Pal et al., 2016). The idea of multi-source encoder is also widely used in the field of APE research (Chatterjee et al., 2018, 2019). In human-machine interaction scenarios, the human feedback is used as extra information in polishing process.

## 6 Conclusion

In this paper, we propose Touch Editing, a flexible and effective interaction approach which allows human translators to revise machine translation results via touch actions. The actions we introduce can be provided with gestures like tapping, panning, swiping or long pressing on touch screens to represent human editing intentions. We present a

simulated action extraction method for constructing training data and a dual-encoder model to handle the actions to generate refined translations.

We prove the effectiveness of the proposed interaction approach and discuss the applicable scenarios with a free-use study. For future works, we plan to conduct large scale real world experiments to evaluate the productivity of different interactive machine translation methods.

## References

- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.
- Sergio Barrachina, Oliver Bender, Francisco Casacuberta, Jorge Civera, Elsa Cubel, Shahram Khadivi, Antonio Lagarda, Hermann Ney, Jesús Tomás, Enrique Vidal, et al. 2009. Statistical approaches to computer-assisted translation. *Computational Linguistics*, 35(1):3–28.
- Ondřej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Hervé Saint-Amand, et al. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the ninth workshop on statistical machine translation*, pages 12–58.
- Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, et al. 2016. Findings of the 2016 conference on machine translation. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 131–198.
- Léon Bottou. 1991. Stochastic gradient learning in neural networks. *Proceedings of Neuro-Nîmes*, 91(8):12.
- Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. 2014. Report on the 11th iwslt evaluation campaign, iwslt 2014. In *Proceedings of the International Workshop on Spoken Language Translation, Hanoi, Vietnam*, volume 57.
- Rajen Chatterjee, Christian Federmann, Matteo Negri, and Marco Turchi. 2019. Findings of the wmt 2019 shared task on automatic post-editing. In *Proceedings of the Fourth Conference on Machine Translation (Volume 3: Shared Task Papers, Day 2)*, pages 11–28.
- Rajen Chatterjee, Matteo Negri, Raphael Rubino, and Marco Turchi. 2018. Findings of the WMT 2018 shared task on automatic post-editing. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*.
- Shanbo Cheng, Shujian Huang, Huadong Chen, Xinyu Dai, and Jiajun Chen. 2016. PRIMT: A pick-revise framework for interactive machine translation. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 1240–1249.
- Michael Denkowski, Chris Dyer, and Alon Lavie. 2014. Learning from post-editing: Online model adaptation for statistical machine translation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 395–404.
- Miguel Domingo, Alvaro Peris, and Francisco Casacuberta. 2016. Interactive-predictive translation based on multiple word-segments. In *Proceedings of the 19th Annual Conference of the European Association for Machine Translation*, pages 282–291.
- Marcello Federico, Alessandro Cattelan, and Marco Trombetti. 2012. Measuring user productivity in machine translation enhanced computer assisted translation. In *Proceedings of the Tenth Conference of the Association for Machine Translation in the Americas (AMTA)*, pages 44–56. AMTA Madison, WI.
- Xinwei Geng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2018. Adaptive multi-pass decoder for neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 523–532.
- Jesús González-Rubio, Daniel Ortiz Martínez, Francisco Casacuberta, and Jose Miguel Benedí Ruiz. 2016. Beyond prefix-based interactive translation prediction. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 198–207.
- David Grangier and Michael Auli. 2018. Quicredit: Editing text & translations by crossing words out. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 272–282.
- Spence Green, Sida I Wang, Jason Chuang, Jeffrey Heer, Sebastian Schuster, and Christopher D Manning. 2014. Human effort and machine learnability in computer aided translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1225–1236.
- Nico Herbig, Tim Düwel, Santanu Pal, Kalliopi Meladaki, Mahsa Monshizadeh, Antonio Krüger, and Josef van Genabith. 2020. Mmpe: A multimodal interface for post-editing machine translation. In *Proceedings of the 58th Annual Meeting of the*

- Association for Computational Linguistics*, pages 1691–1702.
- Nico Herbig, Santanu Pal, Josef van Genabith, and Antonio Krüger. 2019. Multi-modal approaches for post-editing machine translation. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 1–11.
- Chris Hokamp and Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1535–1546.
- Guoping Huang, Jiajun Zhang, Yu Zhou, and Chengqing Zong. 2015. A new input method for human translators: integrating machine translation effectively and imperceptibly. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.
- Hideki Isozaki, Tsutomu Hirao, Kevin Duh, Katsuhito Sudoh, and Hajime Tsukada. 2010. Automatic evaluation of translation quality for distant language pairs. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 944–952. Association for Computational Linguistics.
- Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Rebecca Knowles and Philipp Koehn. 2016. Neural interactive translation prediction. In *Proceedings of the Association for Machine Translation in the Americas*, pages 107–120.
- A-L Lagarda, Vicente Alabau, Francisco Casacuberta, Roberto Silva, and Enrique Díaz-de Liaño. 2009. Statistical post-editing of a rule-based machine translation system. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 217–220. Association for Computational Linguistics.
- Daniel Lopresti and Andrew Tomkins. 1997. Block edit models for approximate string matching. *Theoretical computer science*, 181(1):159–179.
- Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.
- Benjamin Marie and Aurélien Max. 2015. Touch-based pre-post-editing of machine translation output. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1040–1045.
- Bojar Ondrej, Rajen Chatterjee, Federmann Christian, Graham Yvette, Haddow Barry, Huck Matthias, Koehn Philipp, Liu Qun, Logacheva Varvara, Monz Christof, et al. 2017. Findings of the 2017 conference on machine translation (wmt17). In *Second Conference on Machine Translation*, pages 169–214. The Association for Computational Linguistics.
- Santanu Pal, Sudip Kumar Naskar, Mihaela Vela, and Josef van Genabith. 2016. A neural network based approach to automatic post-editing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 281–286.
- Álvaro Peris, Miguel Domingo, and Francisco Casacuberta. 2017. Interactive neural machine translation. *Computer Speech & Language*, 45:201–220.
- Pavel Petrushkov, Shahram Khadivi, and Evgeny Matusov. 2018. Learning from chunk-based feedback in neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 326–331.
- Mirko Plitt and François Masselot. 2010. A productivity test of statistical machine translation post-editing in a typical localisation context. *The Prague bulletin of mathematical linguistics*, 93:7–16.
- Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.
- Dana Shapira and James A Storer. 2002. Edit distance with move operations. In *Annual Symposium on Combinatorial Pattern Matching*, pages 85–98. Springer.
- Michel Simard, Cyril Goutte, and Pierre Isabelle. 2007. Statistical phrase-based post-editing.
- Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of association for machine translation in the Americas*, volume 200.
- Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.
- Gongbo Tang, Mathias Müller, Annette Rios, and Rico Sennrich. 2018a. Why self-attention? a targeted evaluation of neural machine translation architectures. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4263–4272.
- Gongbo Tang, Rico Sennrich, and Joakim Nivre. 2018b. An analysis of attention mechanism: The case of word sense disambiguation in neural machine translation. In *Third Conference on Machine Translation*, pages 26–35.

Amirhossein Tebbifakhr, Ruchit Agrawal, Matteo Negri, and Marco Turchi. 2018. Multi-source transformer with combined losses for automatic post editing. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 846–852.

Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 76–85.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Rongxiang Weng, Hao Zhou, Shujian Huang, Lei Li, Yifan Xia, and Jiajun Chen. 2019. Correct-and-memorize: Learning to translate from interactive revisions. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5255–5263.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google’s neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Joern Wuebker, Spence Green, John DeNero, Sasa Hasan, and Minh-Thang Luong. 2016. Models and inference for prefix-constrained machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 66–75.

Yingce Xia, Fei Tian, Lijun Wu, Jianxin Lin, Tao Qin, Nenghai Yu, and Tie-Yan Liu. 2017. Deliberation networks: Sequence generation beyond one-pass decoding. In *Advances in Neural Information Processing Systems*, pages 1784–1794.

# Can Monolingual Pretrained Models Help Cross-Lingual Classification?

Zewen Chi<sup>†\*</sup>, Li Dong<sup>‡</sup>, Furu Wei<sup>‡</sup>, Xian-Ling Mao<sup>†</sup>, Heyan Huang<sup>†</sup>

<sup>†</sup>Beijing Institute of Technology

<sup>‡</sup>Microsoft Research

{czw,maoxl,hhy63}@bit.edu.cn

{lidong1,fuwei}@microsoft.com

## Abstract

Multilingual pretrained language models (such as multilingual BERT) have achieved impressive results for cross-lingual transfer. However, due to the constant model capacity, multilingual pre-training usually lags behind the monolingual competitors. In this work, we present two approaches to improve zero-shot cross-lingual classification, by transferring the knowledge from monolingual pretrained models to multilingual ones. Experimental results on two cross-lingual classification benchmarks show that our methods outperform vanilla multilingual fine-tuning.

## 1 Introduction

Supervised text classification heavily relies on manually annotated training data, while the data are usually only available in rich-resource languages, such as English. It requires great effort to make the resources available in other languages. Various methods have been proposed to build cross-lingual classification models by exploiting machine translation systems (Xu and Yang, 2017; Chen et al., 2018; Conneau et al., 2018), and learning multilingual embeddings (Conneau et al., 2018; Yu et al., 2018; Artetxe and Schwenk, 2019; Eisenschlos et al., 2019).

Recently, multilingual pretrained language models have shown surprising cross-lingual effectiveness on a wide range of downstream tasks (Devlin et al., 2019; Conneau and Lample, 2019; Conneau et al., 2020; Chi et al., 2020a,b). Even without using any parallel corpora, the pretrained models can still perform zero-shot cross-lingual classification (Pires et al., 2019; Wu and Dredze, 2019; Keung et al., 2019). That is, these models can be fine-tuned in a source language, and then directly evaluated in other target languages. Despite

the effectiveness of cross-lingual transfer, the multilingual pretrained language models have their own drawbacks. Due to the constant number of model parameters, the model capacity of the rich-resource languages decreases if we adds languages for pre-training. The curse of multilinguality results in that the multilingual models usually perform worse than their monolingual competitors on downstream tasks (Arivazhagan et al., 2019; Conneau et al., 2020). The observations motivate us to leverage monolingual pretrained models to improve multilingual models for cross-lingual classification.

In this paper, we propose a multilingual fine-tuning method (MONOX) based on the teacher-student framework, where a multilingual student model learns end task skills from a monolingual teacher. Intuitively, monolingual pretrained models are used to provide supervision of downstream tasks, while multilingual models are employed for knowledge transfer across languages. We conduct experiments on two widely used cross-lingual classification datasets, where our methods outperform baseline models on zero-shot cross-lingual classification. Moreover, we show that the monolingual teacher model can help the student multilingual model for both the source language and target languages, even though the student model is only trained in the source language.

## 2 Background: Multilingual Fine-Tuning

We use multilingual BERT (Devlin et al., 2019) for multilingual pretrained language models. The pretrained model uses the BERT-style Transformer (Vaswani et al., 2017) architecture, and follows the similar fine-tuning procedure as BERT for text classification, which is illustrated in Figure 1(a). To be specific, the first input token of the models is always a special classification token

\*Contribution during internship at Microsoft Research.

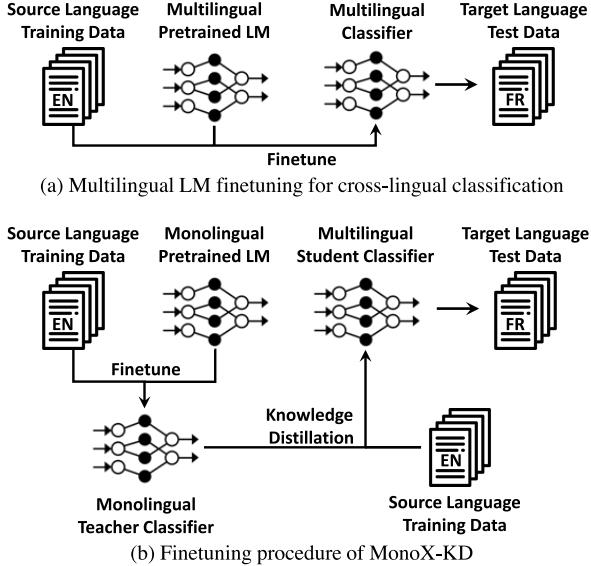


Figure 1: Illustration of multilingual LM fine-tuning. (a) The original multilingual LM fine-tuning procedure for cross-lingual classification. (b) The fine-tuning procedure of our proposed MONOX via knowledge distillation (MONOX-KD). Notice that MONOX does not use any target language data during fine-tuning.

[CLS]. During fine-tuning, the final hidden state of the special token is used as the sentence representation. In order to output predictions, an additional softmax classifier is built on top of the sentence representation. Denoting  $\mathcal{D}$  as the training data in the source language, the pretrained models are fine-tuned with standard cross-entropy loss:

$$\mathcal{L}_{CE}(\theta; \mathcal{D}) = - \sum_{(x,y) \in \mathcal{D}} \log p(y|x; \theta)$$

where  $\theta$  represents model parameters. Then the model is directly evaluated on other languages for cross-lingual classification.

### 3 Methods

As shown in Figure 1(b), we first fine-tune the monolingual pretrained model in the source language. Then we transfer task knowledge to the multilingual pretrained model by soft (Section 3.1) or hard (Section 3.2) labels. We describe two variants of our proposed method (MONOX) as follows.

#### 3.1 Knowledge Distillation

In order to transfer task-specific knowledge from monolingual model to multilingual model, we propose to use knowledge distillation (Hinton et al.,

2015) under our MONOX framework, where a student model  $s$  is trained with soft labels generated by a better-learned teacher model  $t$ . The loss function of the student model is:

$$\begin{aligned} \mathcal{L}_{KD}(\theta_s; \mathcal{D}, \theta_t) = \\ - \sum_{(x,y) \in \mathcal{D}} \sum_{k=1}^K q(y=k|x; \theta_t) \log p(y=k|x; \theta_s) \end{aligned}$$

where  $p(\cdot)$  and  $q(\cdot)$  represent the probability distribution over  $K$  categories, predicted by the student  $s$  and the teacher  $t$ , respectively. Notice that only the student model parameters  $\theta_s$  are updated during knowledge distillation. As shown in Figure 1(b), we first use the fine-tuned monolingual pretrained model as a teacher, which is learned by minimizing  $\mathcal{L}_{CE}(\theta_t; \mathcal{D})$ . Then we perform knowledge distillation for the student model with  $\mathcal{L}_{KD}(\theta_s; \mathcal{D}_C, \theta_t)$  as the loss function, where  $\mathcal{D}_C$  is the concatenation of training dataset and the unlabeled dataset in the source language. We denote this implementation as MONOX-KD.

#### 3.2 Pseudo-Label

In addition to knowledge distillation, we also consider implementing MONOX by training the student multilingual model with pseudo-label (Lee, 2013). Specifically, after fine-tuning the monolingual pretrained model on the training data as teacher, we apply the teacher model on the unlabeled data in the source language to generate pseudo labels. Next, we filter the pseudo labels by a prediction confidence threshold, and only keep the examples with higher confidence scores. Notice that the pseudo training data are assigned with hard labels. Finally, we concatenate the original training data and the pseudo data as the final training set for the student model. We denote this implementation as MONOX-PL.

## 4 Experiments

### 4.1 Experimental Setup

In the following experiments, we consider the zero-shot cross-lingual setting, where models are trained with English data and directly evaluated on all target languages.

**Datasets** We conduct experiments on two widely used datasets for cross-lingual evaluation: (1) Cross-Lingual Sentiment (CLS) dataset (Prettenhofer and Stein, 2010), containing Amazon

reviews in three domains and four languages; (2) Cross-Lingual NLI (XNLI) dataset (Conneau et al., 2018), containing development and test sets in 15 languages and a training set in English for the natural language inference task.

**Pretrained Language Models** We use multilingual BERT<sub>BASE</sub><sup>1</sup> for cross-lingual transfer. For monolingual pretrained language model, the English-version RoBERTa<sub>LARGE</sub><sup>2</sup> is employed. All the pretrained models used in our experiments are cased models.

**Baselines** We compare our methods (MONOX-KD, and MONOX-PL) with the following models:

- MBERT: directly fine-tuning the multilingual BERT<sub>BASE</sub> with English training data.
- MBERT-ST: fine-tuning the multilingual BERT<sub>BASE</sub> by self-training, i.e., alternately fine-tuning mBERT and updating the training data by labeling English unlabeled examples.

## 4.2 Configuration

For the CLS dataset, we randomly select 20% examples from training data as the development set and use the remaining examples as the training set. For XNLI, we randomly sample 20% examples from training data as the training set, and regard the other examples as the unlabeled set. We use the vocabularies provided by the pretrained models, which are extracted by Byte-Pair Encoding (Sennrich et al., 2016). The input sentences are truncated to 256 tokens. For both datasets, we use Adam optimizer with a learning rate of  $5 \times 10^{-6}$ , and a batch size of 8. We train models with epoch size of 200 and 2,500 steps for CLS and XNLI, respectively. For MONOX-KD, the softmax temperature of knowledge distillation is set to 0.1. For MONOX-PL, the confidence threshold is set to zero, which means all of the generated pseudo labels are used as training data.

## 4.3 Results and Discussion

**Preliminary Experiments** To see how much monolingual pretrained models is better than multilingual pretrained models, we finetune several different pretrained language models on the two datasets under the aforementioned configuration,

<sup>1</sup><https://github.com/google-research/bert/blob/master/multilingual.md>

<sup>2</sup><https://github.com/pytorch/fairseq/tree/master/examples/roberta>

	Parameters	CLS	XNLI
<i>Multilingual Pretrained Models</i>			
MBERT	110M	86.37	77.07
<i>Monolingual Pretrained Models</i>			
BERT <sub>BASE</sub>	110M	90.10	80.46
RoBERTa <sub>BASE</sub>	125M	93.82	85.09
RoBERTa <sub>LARGE</sub>	355M	95.77	89.24

Table 1: Preliminary experiments results. Models are finetuned with English training data of CLS and XNLI under the configuration (see Section 3.2), and only evaluated in English. The results on CLS are averaged over three domains.

and only evaluate them in English. As shown in Table 1, the gap between multilingual and monolingual pretrained models is large, even when using the same size of parameters. It is not hard to explain because MBERT is trained in 104 languages, where different languages tend to confuse each other.

**Sentiment Classification** We evaluate our method on the zero-shot cross-lingual sentiment classification task. The goal of sentiment classification is to classify input sentences to positive or negative sentiments. In Table 2 we compare the results of our methods with baselines on CLS. It can be observed that our MONOX method outperforms baselines in all evaluated languages and domains, providing 4.91% improvement of averaged accuracy to the original multilingual BERT fine-tuning method. Notice that MBERT-ST is trained under the same condition with our method, i.e., using the same labeled and unlabeled data as ours. However, we only observe a slight improvement over MBERT, which demonstrates that the performance improvement of MONOX mainly benefits from its end task knowledge transfer rather than the unlabeled data.

**Natural Language Inference** We also evaluate our method on the zero-shot cross-lingual NLI task, which is more challenging than sentiment classification. The goal of NLI is to identify the relationship of a pair of input sentences, including a premise and a hypothesis with an *entailment*, *contradiction*, or *neutral* relationship between them. As shown in Table 3, we present the evaluation results on XNLI. Unsurprisingly, both MONOX-PL and MONOX-KD perform better than baseline methods, showing that our method success-

	Books	en DVD	Music	Books	de DVD	Music	Books	fr DVD	Music	Books	ja DVD	Music	avg
MBERT	87.75	86.60	84.75	79.55	75.90	77.05	81.45	80.35	80.35	75.15	76.90	75.90	80.14
MBERT-ST	88.20	85.50	88.00	79.65	76.70	80.00	84.85	83.25	80.55	74.60	75.80	76.90	81.17
MONOX-PL	<b>94.00</b>	<b>92.75</b>	91.80	83.20	79.25	82.95	<b>86.00</b>	84.95	<b>84.55</b>	78.85	<b>80.00</b>	79.35	84.80
MONOX-KD	93.90	91.40	<b>92.25</b>	<b>84.20</b>	<b>81.50</b>	<b>83.65</b>	85.40	<b>85.90</b>	83.95	<b>78.95</b>	79.15	<b>80.30</b>	<b>85.05</b>

Table 2: Evaluation results of zero-shot cross-lingual sentiment classification on the CLS dataset.

	ar	bg	de	el	en	es	fr	hi	ru	sw	th	tr	ur	vi	zh	avg
MBERT	61.2	67.4	65.8	61.6	77.1	70.7	68.6	<b>53.4</b>	67.0	50.6	<b>44.6</b>	56.3	57.8	43.6	67.8	60.9
MBERT-ST	60.9	67.6	65.4	61.0	77.6	70.4	68.9	53.1	65.9	50.6	41.8	55.2	56.8	43.6	67.9	60.5
MONOX-PL	<b>63.5</b>	<b>70.1</b>	<b>69.8</b>	61.7	<b>80.9</b>	<b>74.1</b>	<b>72.1</b>	52.5	68.4	51.2	42.3	<b>57.9</b>	58.0	44.0	70.2	<b>62.5</b>
MONOX-KD	62.2	69.3	69.3	<b>62.1</b>	79.6	72.9	72.0	52.8	<b>68.6</b>	<b>52.3</b>	41.7	<b>57.9</b>	<b>58.5</b>	<b>45.9</b>	<b>70.8</b>	62.4

Table 3: Evaluation results of zero-shot cross-lingual NLI on the XNLI dataset. Note that 20% of the original training data are used as training set, and the other 80% are used as unlabeled set.

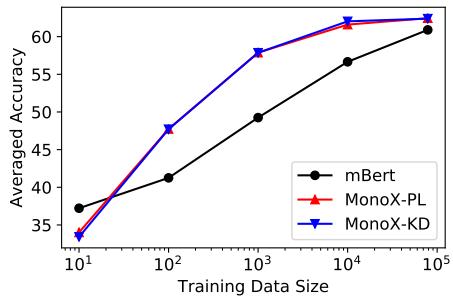


Figure 2: Averaged accuracy scores on zero-shot XNLI with different training data sizes. (20% and 80% of the training data are regraded training and unlabeled set.)

fully helps the multilingual pretrained model gain end task knowledge from the monolingual pretrained model for cross-lingual classification. It is also worth mentioning that the performance of MBERT-ST is similar to MBERT. We believe the reason is that XNLI has more training data than CLS, which weakens the impact of self-training.

**Effects of Training Data Size** We conduct a study on how much multilingual pretrained model can learn from monolingual pretrained model for different training data size. We cut the training data to 10, 100, 1K, 10K and 78K (full training data in our setting) examples, and keep other hyper-parameters fixed. In Figure 2, we show the averaged accuracy scores for zero-shot XNLI with different training data sizes. We observe that MONOX outperforms MBERT on all data sizes except the 10-example setting. When the training data is relatively small ( $\leq 10^4$ ), our method shows

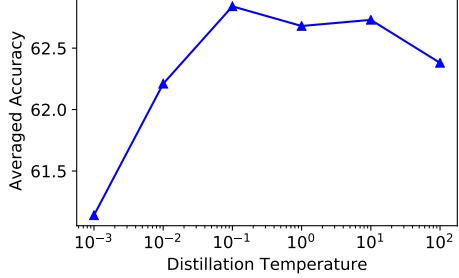


Figure 3: Averaged accuracy scores on the development set for zero-shot XNLI with different softmax temperatures of MONOX-KD.

a great improvement.

**Effects of Distillation Temperature** Figure 3 presents XNLI averaged accuracy scores of MONOX-KD with different softmax temperatures in knowledge distillation. Even though the temperature varies from  $10^{-3}$  to  $10^2$ , all of the results are higher than baseline scores, which indicates MONOX-KD is nonsensitive to the temperature. When the temperature is set to  $10^{-1}$ , we observe the best results on the development set. Therefore we set temperature as 0.1 in other experiments.

## 5 Conclusion

In this work, we investigated whether a monolingual pretrained model can help cross-lingual classification. Our results have shown that, with a RoBERTa model pretrained in English, we can boost the classification performance of a pretrained multilingual BERT in other languages. For future work, we will explore whether mono-

lingual pretrained models can help other cross-lingual NLP tasks, such as natural language generation (Chi et al., 2020a).

## Acknowledgements

Prof. Heyan Huang is the corresponding author. The work is supported by National Key R&D Plan (No. 2016QY03D0602), NSFC (No. U19B2020, 61772076, 61751201 and 61602197) and NSFB (No. Z181100008918002).

## References

- Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, M. Gut Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George Foster, Colin Cherry, Wolfgang Macherey, Zhifeng Chen, and Yonghui Wu. 2019. Massively multilingual neural machine translation in the wild: Findings and challenges. *ArXiv*, abs/1907.05019.
- Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.
- Xilun Chen, Yu Sun, Ben Athiwaratkun, Claire Cardie, and Kilian Weinberger. 2018. Adversarial deep averaging networks for cross-lingual sentiment classification. *Transactions of the Association for Computational Linguistics*, 6:557–570.
- Zewen Chi, Li Dong, Furu Wei, Wenhui Wang, Xian-Ling Mao, and Heyan Huang. 2020a. Cross-lingual natural language generation via pre-training. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 7570–7577. AAAI Press.
- Zewen Chi, Li Dong, Furu Wei, Nan Yang, Saksham Singhal, Wenhui Wang, Xia Song, Xian-Ling Mao, Heyan Huang, and Ming Zhou. 2020b. InfoXML: An information-theoretic framework for cross-lingual language model pre-training. *ArXiv*, abs/2007.07834.
- Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.
- Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems*, pages 7057–7067. Curran Associates, Inc.
- Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- Julian Eisenschlos, Sebastian Ruder, Piotr Czapla, Marcin Kadras, Sylvain Gugger, and Jeremy Howard. 2019. MultiFiT: Efficient multi-lingual language model fine-tuning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5702–5707, Hong Kong, China. Association for Computational Linguistics.
- Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. *ArXiv*, abs/1503.02531.
- Phillip Keung, Yichao Lu, and Vikas Bhardwaj. 2019. Adversarial learning with contextual embeddings for zero-resource cross-lingual classification and NER. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1355–1360, Hong Kong, China. Association for Computational Linguistics.
- Dong-Hyun Lee. 2013. Pseudo-label : The simple and efficient semi-supervised learning method for deep neural networks. *ICML 2013 Workshop : Challenges in Representation Learning*.
- Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.
- Peter Prettenhofer and Benno Stein. 2010. Cross-language text classification using structural correspondence learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1118–1127, Uppsala, Sweden. Association for Computational Linguistics.
- Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational*

*Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. [Attention is all you need](#). In *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Shijie Wu and Mark Dredze. 2019. [Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 833–844, Hong Kong, China. Association for Computational Linguistics.

Ruochen Xu and Yiming Yang. 2017. [Cross-lingual distillation for text classification](#). In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1415–1425, Vancouver, Canada. Association for Computational Linguistics.

Katherine Yu, Haoran Li, and Barlas Ouz. 2018. Multilingual seq2seq training with similarity loss for cross-lingual document classification. In *Proceedings of The Third Workshop on Representation Learning for NLP*, pages 175–179.

# Rumor Detection on Twitter Using Multiloss Hierarchical BiLSTM with an Attenuation Factor

Yudianto Sujana\*, Jiawen Li\*, Hung-Yu Kao

Intelligent Knowledge Management Lab

Department of Computer Science and Information Engineering

National Cheng Kung University

Tainan, Taiwan

yudianto.sujana@staff.uns.ac.id, p78073012@gs.ncku.edu.tw,  
hykao@mail.ncku.edu.tw

## Abstract

Social media platforms such as Twitter have become a breeding ground for unverified information or rumors. These rumors can threaten people's health, endanger the economy, and affect the stability of a country. Many researchers have developed models to classify rumors using traditional machine learning or vanilla deep learning models. However, previous studies on rumor detection have achieved low precision and are time consuming. Inspired by the hierarchical model and multitask learning, a multiloss hierarchical BiLSTM model with an attenuation factor is proposed in this paper. The model is divided into two BiLSTM modules: post level and event level. By means of this hierarchical structure, the model can extract deep information from limited quantities of text. Each module has a loss function that helps to learn bilateral features and reduce the training time. An attenuation factor is added at the post level to increase the accuracy. The results on two rumor datasets demonstrate that our model achieves better performance than that of state-of-the-art machine learning and vanilla deep learning models.

## 1 Introduction

Currently, social media has a significant influence on people's daily lives. With social media, people can share information, speak freely and reproduce news online conveniently. Take Twitter as an example: over 500 million new tweets are sent every day, that is, nearly 5787 tweets per second (Cooper, 2019). However, unverified information, or rumors, is also diffused in social media; therefore, in the absence of a rumor detection

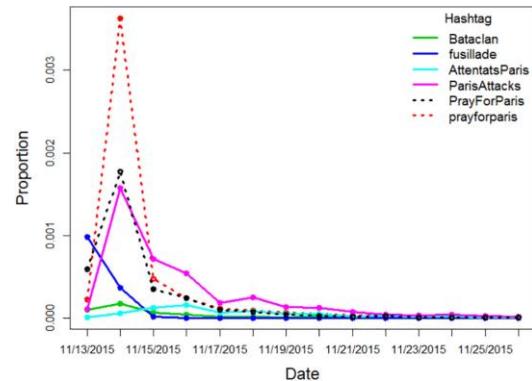


Figure 1: Relationship between the popularity of relevant hashtags about the Paris attack and time (Cvetojevic and Hochmair, 2018).

system, social media platforms can become a breeding ground for rumors.

In 2013, the Associated Press's official Twitter account was hacked and sent out a rumor that the president of the US was injured in an attack. This rumor caused wide panic and resulted in a brief crash of the stock market, in which investors lost \$136 billion in just two minutes (Aja et al., 2018). This incident highlighted that misinformation can threaten people's lives. Therefore, an automatic rumor detection system is vital. Information popularity in social media has a short lifetime; it usually stays for a few days, which is called the explosive increase phase. For instance, in Figure 1, tweets related to the Paris attack only stayed popular for two days (Cvetojevic and Hochmair, 2018). Unfortunately, Vosoughi et al. (2018) confirmed that false information propagated faster and was longer lasting than true information. Their research shows that it took nearly six times longer for verified information to reach 1500 people than for a rumor. According to the study, early as possible detection is highly practical to minimize harmful effects before rumors enter into the

\* Equal contributions.

explosive increase phase. However, early rumor detection is the most difficult component of overall rumor detection. The greatest challenge lies in the lack of information.

To solve rumor detection problems, we analyze comments on a post. Comments can help in self-correcting the information dissemination through opinions, guesses and evidence shared by users. Thus, readers can judge the authenticity of information (Zubiaga et al., 2018). When commenting on an unverified post, people were inclined to use an interrogative or rhetorical tone (Kim, 2014; Ma et al., 2018a). Furthermore, rumors could be detected via the route of information diffusion(Ma et al., 2018b). However, the number of comments on a post is sometimes too narrow to use in early rumor detection. Therefore, making full use of limited information for accurate judgment remains a formidable challenge.

One important question is how to use such information. Zubiaga et al. (2016, 2017) proposed a method to use this information as a context to determine whether a tweet constituted a rumor. In contrast, some scholars suggested that events could be utilized as the basic processing unit for rumor detection, such as the tree-structured recursive neural network (Ma et al., 2018b), hierarchical structure model (Guo et al., 2018), and multitask learning (Ma et al., 2018a). Generally, an event contains an original post and a series of replies. Most of the scholars mentioned above use a large number of replies (from hundreds to thousands) to assist in detection. However, we believe that a large number of comments is not in line with the goal of early rumor detection; therefore, only one original post and a few early replies are used in this paper. Considering that the performance and capacity of a single processing layer to fully extract the text information is poor, we assume that higher-level structural models will bring more benefits for detection.

We attempted to represent information through a hierarchical neural network by building a post-level module first and an event-level module based on it. Since post-based and event-based rumor detection are highly related tasks, the hierarchical structure model can easily learn the bilateral feature representation based on these two tasks. In contrast to the traditional hierarchical structure, the model is based on a bidirectional long short-term memory (BiLSTM) model with some

improvements. By means of the concept of multitask learning, we established a hierarchical model with a multiloss function to shorten the model training time and added an attenuation factor to the post-level model to maintain its precision. With this structure, the model can alleviate the impact of the vanishing gradient problem to a certain extent. The experimental results show that our model outperforms current state-of-the-art models.

Our contributions to this topic are as follows: (1) To the best of our knowledge, this is the first time that a multiloss function model with an attenuation factor was used for rumor detection. The model successfully combines post-level and event-level information for rumor detection. (2) The results of an evaluation using actual data from Twitter show that our model achieved high accuracy with only a few posts.

## 2 Related Work

Current automatic rumor detection systems suffer from low accuracy (Zubiaga et al., 2018). Two main approaches are used to debunk misinformation: the traditional method and the artificial intelligence approach. The traditional method manually analyzes text using statistics to define the critical features before detection. Castillo et al. (2011) proposed a large number of features for rumor detection by analyzing user attributes, rumor diffusion routes and text. Some researchers introduce various sets of features from different perspectives (Liu et al., 2015; K. Wu et al., 2015; Yang et al., 2012). With the development of artificial intelligence, some scholars have attempted to recognize rumors using deep learning. Ma et al. (2016) and Rath et al. (2017) used the RNN model to learn the abstract expression of rumors. Guo et al. (2018) proposed a hierarchical social attention model by combining a deep learning model and feature engineering, which improved the precision of rumor detection.

Early detection is the most challenging part of rumor detection. Many attempts at early rumor detection have been made. Wang et al. (2017) analyzed prominent features of rumor propagation and proposed a probabilistic model. Kwon et al. (2017) found that user and linguistic features could be used as important indicators in rumor detection. In addition to traditional methods, machine learning and deep learning have been applied to early rumor detection. Wu et al. (2015) proposed a

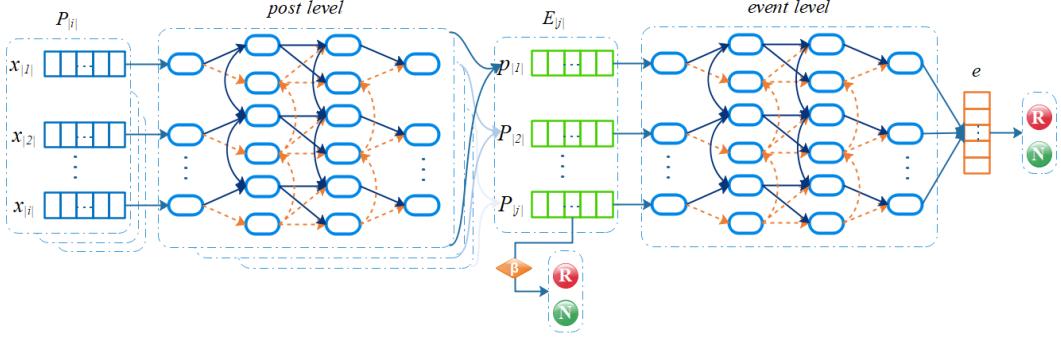


Figure 2: Multiloss BiLSTM hierarchical structure model with an attenuation factor.

graph-kernel-based hybrid SVM classifier that could capture high-order propagation patterns. Zhao et al. (2015) developed a technique based on searching for enquiry phrases that yielded good performance. Zhang et al. (2018) proposed a heterogeneous network for early rumor detection that reached 61% precision. T. Chen et al. (2018) used an RNN network with soft-attention structures. L. Wu et al. (2017) built a neural network framework consisting of inferring rumor categories, selecting discriminative features, and learning a rumor classifier. Moreover, Nguyen et al. (2017) presented an approach that leveraged convolutional neural networks for learning the hidden representations of each tweet in combination with a time series.

Inspired by multitask learning and the hierarchical structure, we developed a multiloss hierarchical BiLSTM model with an attenuation factor that has high accuracy and performs well in early rumor detection.

### 3 Problem Formulation

A tweet consists of a limited number of words, some emojis and a few hashtags. This limited text makes it hard to classify misinformation. Therefore, we consider combining the source tweet and some of its comments as a whole event for rumor detection. We employ an event as the primary processing unit. An event contains more information and implicit users' stance (Liu et al., 2015; Lukasik et al., 2015, 2016; Mendoza et al., 2010; Rosengren et al., 2011).

Our hierarchical structure model begins with word embedding followed by post embedding and event embedding, with a fully connected layer at the end to detect whether the event is a rumor.

Multiple topics in the dataset are defined as  $T = \{T_1, T_2, \dots, T_{|t|}\}$ , and each  $T_i$  contains multiple

events,  $T_i = \{E_1, E_2, \dots, E_{|e|}\}$ . An event consists of a source post and a few comments,  $E_e = \{P_s, P_1, P_2, \dots, P_{|p|}\}$ . Notably, different topics have a different number of events, and events contain different numbers of posts, which means our model can handle variable length information. We develop this rumor detection task as a supervised classification problem. The classifier can perform learning via labeled information, that is,  $f_e: \{E_1, E_2, \dots, E_{|e|}\} \rightarrow y_e$ . At the same time, each post has its own label. Here, we define that each post label is identical to its corresponding source post and equivalent to the label of the event. Therefore, the post-level classifier  $f_p: \{P_s, P_1, P_2, \dots, P_{|p|}\} \rightarrow y_p$  can be established, and all labels take one of two possible class labels: rumor or nonrumor.

### 4 Multiloss Hierarchical BiLSTM with an Attenuation Factor

The experimental results show that the hierarchical structure has strong information expression ability. However, our observations indicate that the hierarchical model has certain deficiencies, and backpropagation has to go through the time steps of all previous layers, which is computationally expensive and inefficient. It may also lead to vanishing gradient problems and substantially increase the training time. To shorten the training time and improve the training efficiency, we proposed a multiloss BiLSTM hierarchical structure model. Compared to the regular hierarchical model, this multiloss model is equivalent to a multitask learning model that can benefit bilaterally from the information features among multiple related tasks. Rumor detection at the post level and event level represent two branches under this theme, and the representations learned in the post level can be shared and used to

reinforce the feature learning at the event level. Importantly, the backpropagation of the post level can help to alleviate the vanishing gradient problem in the early stage; thus, the model is stable and the training time is reduced.

Although the multiloss function can accelerate model training, the accuracy is slightly reduced because both post and event factors are considered in the classification process. Therefore, we added an attenuation factor to the post level to decrease the training time and maintain the high accuracy simultaneously.

Taking the text of all posts under an event as the input, we first perform word-embedding processing, where the processed word can be expressed as a fixed-length text vector. The formula is as follows:

$$x_t = E\theta x_t \quad (1)$$

where  $x_t$  is the  $t_{th}$  word in a post and  $E$  is a special word-embedding matrix. This step is omitted from the model diagram.

Next, all the vectors with the post as the unit pass through the post-level BiLSTM layer in proper order. Here, a deep BiLSTM structure is used. For each time point  $t$ , the formula is as follows:

$$h_{POST_i} = BiLSTM(x_i, h_{POST_{i-1}}) \quad (2)$$

According to the physical meaning of LSTM, the cell state  $h_t$  of the uppermost LSTM at the last time point is used as the result of the post encoding. Due to the use of the bidirectional structure, the final state of both directions is jointed, and an event can be represented by a matrix in which each column is a vector representing a post. The formula is as follows:

$$X = [h_{LSTM_{P_1}}, h_{LSTM_{P_2}}, \dots, h_{LSTM_{P_{|P|}}}] \quad (3)$$

where  $h_{LSTM_{P_i}}$  is the result from the post-level BiLSTM, that is, the embedding of one post.

The event-level BiLSTM formula is similar to the post-level BiLSTM. The difference is the input, where post-level BiLSTM uses  $x_i$  and the event-level BiLSTM uses  $X_I$ :

$$h_{EVENT_i} = BiLSTM(X_I, h_{EVENT_{i-1}}) \quad (4)$$

In the rumor detection classification task, the state of the event-level BiLSTM top layer at the last time point can be understood as an abstract representation of all post understandings.

To shorten the training time, the concept of multitask learning is used as a reference to realize the detection tasks of both posts and events. These two tasks are highly correlated, and the parameters in the post layer can be understood as common features. From the perspective of multitasking, the common feature region can assist the model training for the purpose of rapid convergence.

A post-level classifier and an event-level classifier are included in the model. Note that the post-level classifier functions only as an auxiliary convergence, so the post-level classifier classifies and backpropagates only the last set of posts for each event.

$$\begin{aligned} y_p &= \text{soft max}(W_p * h_{LSTM_{P_{|P|}}} + b_p) \\ y_e &= \text{soft max}(W_e * h_{EVENT_{|P|}} + b_e) \end{aligned} \quad (5)$$

where  $y_p$  and  $y_e$  represent the post and event classification results, respectively,  $W_p$  and  $W_e$  are the weights of the fully connected layers, and  $b_p$  and  $b_e$  are the biases.

The multiloss function helps to achieve rapid convergence, but it reduces the accuracy. To realize the rapid training of the model while maintaining its precision, an attenuation factor is added in the backpropagation. Since the Adam optimizer is used, the formula is as follows:

$$\begin{aligned} g_e &= \nabla_{\theta_{t-1}} f(\theta_{t-1}) \\ m_t &= \mu * m_{t-1} + (1 - \mu) * g_t \\ n_t &= \nu * n_{t-1} + (1 - \nu) * g_t^2 \\ \hat{m}_t &= m_t / (1 - \mu^t) \\ \hat{n}_t &= n_t / (1 - \nu^t) \end{aligned} \quad (6)$$

$$\Delta \theta_t^e = -\eta * \hat{m}_t^e / \sqrt{\hat{n}_t^e} + \varepsilon \quad (7)$$

$$\Delta \theta_t^p = -(\eta * \hat{m}_t^p / \sqrt{\hat{n}_t^p} + \varepsilon) * \beta \quad (8)$$

where  $\hat{m}_t$  and  $\hat{n}_t$  are the corrections of  $m_t$  and  $n_t$ , respectively.  $m_t$  and  $n_t$  are the first-order and second-order moment estimates of the gradient under the event, respectively, which can be regarded as estimates of the expectation and be approximated as unbiased estimates.  $[\mu, \nu, \varepsilon]$  are hyperparameters, and  $\hat{m}_t^e$ ,  $\hat{n}_t^e$ , and  $\theta_t^e$  represent the corresponding parameters of an event.  $\beta$  is an attenuation factor, which decreases to zero as the number of training epochs increases.

## 5 Experiments and Results

### 5.1 Data Collection

The data from two rumor datasets used in this study are derived from tweets posted during breaking news. Table 2 describes the statistics of these two datasets. Moreover, the two datasets contain a large number of properties that can be used for feature engineering, which is helpful for rumor detection. However, since we build a model based on deep learning, the model learns the features automatically from the posts.

Statistic	PHEME	
	2017	2018
Users	49,345	50,593
Posts	103,212	105,354
Events	5,802	6,425
Avg words/post	13.6	13.6
Avg posts/event	17.8	16.3
Max posts/event	346	246
Rumor	1972	2402
Nonrumor	3830	4023
Balance degree	34.00%	37.40%

Table 2: Dataset statistics

### 5.2 Model Training

For our experiment, the datasets were randomly split: 80% for training, 10% for validation, and 10% for testing. Similar to the work of Ma et al. (2016), we calculated the accuracy, precision, recall and F1-score to measure the rumor detection performance.

In the data preprocessing phase, our data were subjected to the following processes: standardizing text and deleting useless network labels, emojis, etc. However, the stop words were retained because they contain words that can be used to reflect the emotions of the writer. We trained all the models by employing the derivative of the loss function through backpropagation and used the Adam optimizer to update the parameters. For the hyperparameters, the maximum value of vocabulary is 25000, the batch size is 64, the dropout rate is 0.5, the hidden size unit is 256, and the learning rate is 0.0001. Training was then performed based on different events until the loss value converged or the maximum number of epochs was reached.

Dataset	Method	Acc	Pre	Rec	F1
2017	SVM-BOW	0.669	0.535	0.524	0.529
	CNN	0.787	0.737	0.702	0.719
	BiLSTM	0.795	0.763	0.691	0.725
	BERT	0.865	<b>0.859</b>	0.851	0.855
	RDM*	0.873	0.817	0.823	0.820
	<b>MHA</b>	<b>0.926</b>	0.834	<b>0.956</b>	<b>0.891</b>
2018	SVM-BOW	0.688	0.518	0.512	0.515
	CNN	0.795	0.731	0.673	0.701
	BiLSTM	0.794	0.727	0.677	0.701
	BERT	0.844	0.834	0.835	0.834
	RDM*	0.858	0.847	0.859	0.853
	<b>MHA</b>	<b>0.919</b>	<b>0.892</b>	<b>0.923</b>	<b>0.907</b>

Table 1: Comparison results

### 5.3 Result

We compare our model with the following models:

- SVM-BOW: SVM classifier using bag-of-words and N-gram (e.g., 1-gram, bigram and trigram) features (Ma et al., 2018b).
- CNN: A convolutional neural network model (Y.-C. Chen et al., 2017) for obtaining the representation of each tweet and classifying tweets with a softmax layer.
- BiLSTM: A bidirectional RNN-based tweet model (Augenstein et al., 2016) that considers the bidirectional context between the target and tweet.
- BERT: A fine-tuned BERT to detect rumors.
- RDM: A method that integrates GRU and reinforcement learning to detect rumors in the early stage (Zhou et al., 2019).
- MHA: Our hierarchical model with a multiloss function and an attenuation factor.

The results of all the methods are illustrated in Table 1, and the MHA model yields the best performance. The SVM-BOW result is poor because the traditional statistical machine learning method is not able to capture helpful features in this complicated rumor detection task. For the CNN, BiLSTM, and RDM models, the results are worse than those of our model due to the insufficient

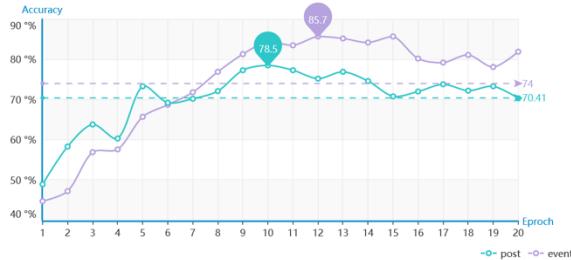


Figure 4: Comparison between post and event-based detection.

capacity for information extraction. Those models process information based on posts and cannot obtain high-level representations from a hierarchical structure. BERT achieves state-of-the-art performance in many other NLP tasks. It has multiple layers and multihead attention and can mine in-depth information, but this structure is also based on posts and does not consider the post-event structure.

#### 5.4 Ablation Experiments

**Event and Post Analysis.** We suggest that rumor detection based on an event is more credible than rumor detection based on a post. To prove this point, we conducted an experiment in which two models with identical structures and parameters were used to detect rumors in two different datasets. These two datasets contain the same text information: the only difference is that one is based on posts and the other is based on events. From the experimental results shown in Figure 4, we can see that the accuracy of the model with the event dataset is approximately 7% higher. This result verifies our assumption that rumor detection with events as the detecting unit is more accurate. Meanwhile, such an idea also paves the way for us to develop the hierarchical structure.

**General Structure and Hierarchical Structure Comparison.** We believe that the hierarchical structure, which has an advanced processing unit, is superior to the general structure in terms of extracting more complex and more in-depth information. To prove our hypothesis, we compared the general BiLSTM with the hierarchical BiLSTM (post-event layer). The hierarchical structure has two levels, namely, the post and event, in which the output from the post level becomes the input of the event level.

We used the same parameters for each module to ensure a fair comparison. Figure 5 shows that the



Figure 5: Comparison between the general BiLSTM model and hierarchical BiLSTM.

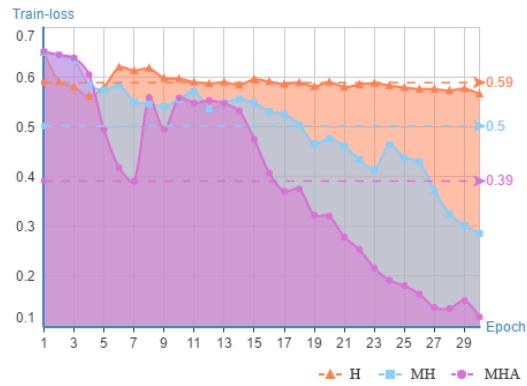


Figure 3: Loss comparison.

hierarchical structure outperforms the general structure in terms of accuracy, precision, recall, and F1, which confirms our hypothesis that the hierarchical structure has stronger detection capability.

**Effects of Multiloss Functions and Attenuation Factor.** We also evaluated several internal models to show how the multiloss function helps in rumor detection and to further investigate the impact of the attenuation factor in the proposed model:

- Hierarchical (H): BiLSTM hierarchical structure model.
- Multiloss Hierarchical (MH): Multiloss BiLSTM hierarchical structure model.
- MHA: Multiloss BiLSTM hierarchical structure model with an attenuation factor

We compared the training results of the H, MH, and MHA models on the same dataset with the same random seed. Figure 3 shows that the MH and MHA, which are multiloss function models, learn faster than the original hierarchy model in the first 30 epochs. Moreover, the loss in that model decreases sharply. This result proves that the

Test dataset	Number of posts in each event
Test_5	2 – 5
Test_10	6 – 10
Test_15	11 – 15
Test_20	16 – 20
Test_25	21 – 25
Test_30	26 – 30

Table 3: Test dataset for early rumor detection

models benefit from post-level backpropagation by applying a multiloss function.

The attenuation factor in MHA gradually decreased to zero until epoch fifteen. This attenuation factor makes the MHA model learn based on only the event label, whereas the HA model continues to tune the parameters based on both post and event information. With this technique, the training process becomes faster while maintaining the loss decreases.

### 5.5 Early Rumor Detection

To evaluate the model’s early rumor detection performance, we considered six types of test sets that reflect the real scenario of rumor spreading on Twitter.

A small number of posts for each event means that the rumor had just begun to spread, with only a few tweets about the rumor. On the other hand, a large number of tweets implies that the rumors have spread widely.

The test results shown in Figure 6 indicate that our MHA model detects rumors better and with higher accuracy in the Test\_5 dataset than do the other methods. This result implies that our models can classify rumors very early. Furthermore, our model also performs well in other test datasets, which indicates that our model can be used to detect both new rumors and widely spread rumors.

## 6 Conclusion and Future Work

In this paper, we introduced a multiloss hierarchical BiLSTM with an attenuation factor model for rumor detection. By means of the hierarchical structure, the model can learn deeply from limited text. The multiloss function makes the model learn efficiently and robustly, while the attenuation factor at the post level helps to increase the accuracy of rumor detection. The experimental results based on two PHEME datasets demonstrate that the model consistently outperforms other models by a significant margin. The model represents any post and event text with a fixed size length vector, which means it has strong

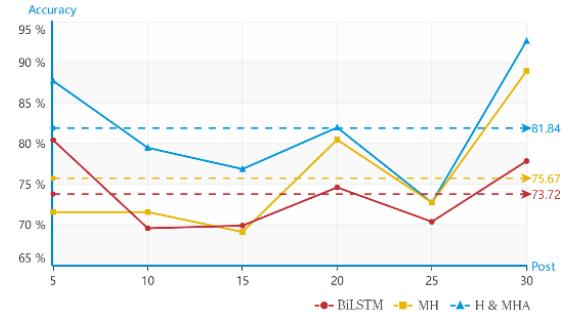


Figure 6: Early rumor detection accuracy at different numbers of posts.

applicability for both early and widely spread rumor detection with only a few modifications. In the future, the model can be extended by implementing social feature engineering to analyze and track rumors.

## References

- Ajao, O., Bhowmik, D., & Zargari, S. 2018. [Fake News Identification on Twitter with Hybrid CNN and RNN Models](#). *Proceedings of the 9th International Conference on Social Media and Society*, 226–230. <https://doi.org/10.1145/3217804.3217917>
- Augenstein, I., Rocktäschel, T., Vlachos, A., & Bontcheva, K. 2016. [Stance Detection with Bidirectional Conditional Encoding](#). *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 35(6), 876–885. <https://doi.org/10.18653/v1/D16-1084>
- Castillo, C., Mendoza, M., & Poblete, B. 2011. [Information credibility on twitter](#). *Proceedings of the 20th International Conference on World Wide Web - WWW '11*, 675. <https://doi.org/10.1145/1963405.1963500>
- Chen, T., Li, X., Yin, H., & Zhang, J. 2018. [Call Attention to Rumors: Deep Attention Based Recurrent Neural Networks for Early Rumor Detection](#). In *Lecture Notes in Computer Science: Vol. 11154 LNAI* (pp. 40–52). [https://doi.org/10.1007/978-3-030-04503-6\\_4](https://doi.org/10.1007/978-3-030-04503-6_4)
- Chen, Y.-C., Liu, Z.-Y., & Kao, H.-Y. 2017. [IKM at SemEval-2017 Task 8: Convolutional Neural Networks for stance detection and rumor verification](#). *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, (2011), 465–469. <https://doi.org/10.18653/v1/S17-2081>
- Cooper, P. 2019. [28 Twitter Statistics All Marketers Need to Know in 2019](#). Retrieved February 18, 2019, from <https://blog.hootsuite.com/twitter-statistics/>
- Cvetojevic, S., & Hochmair, H. H. 2018. [Analyzing the spread of tweets in response to Paris attacks](#).

- Computers, Environment and Urban Systems*, 71(April), 14–26. <https://doi.org/10.1016/j.compenvurbsys.2018.03.010>
- Guo, H., Cao, J., Zhang, Y., Guo, J., & Li, J. 2018. Rumor Detection with Hierarchical Social Attention Network. *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 943–951. <https://doi.org/10.1145/3269206.3271709>
- Kim, Y. 2014. Convolutional Neural Networks for Sentence Classification. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 218, 1746–1751. <https://doi.org/10.3115/v1/D14-1181>
- Kwon, S., Cha, M., & Jung, K. 2017. Rumor Detection over Varying Time Windows. *PLOS ONE*, 12(1), 1–19. <https://doi.org/10.1371/journal.pone.0168344>
- Liu, X., Nourbakhsh, A., Li, Q., Fang, R., & Shah, S. 2015. Real-time Rumor Debunking on Twitter. *Proceedings of the 24th ACM International Conference on Information and Knowledge Management - CIKM '15*, 1867–1870. <https://doi.org/10.1145/2806416.2806651>
- Lukasik, M., Cohn, T., & Bontcheva, K. 2015. Classifying Tweet Level Judgements of Rumours in Social Media. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, (September), 2590–2595. <https://doi.org/10.18653/v1/D15-1311>
- Lukasik, M., Srijith, P. K., Vu, D., Bontcheva, K., Zubiaga, A., & Cohn, T. 2016. Hawkes Processes for Continuous Time Sequence Classification: an Application to Rumour Stance Classification in Twitter. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 393–398. <https://doi.org/10.18653/v1/P16-2064>
- Ma, J., Gao, W., Mitra, P., Kwon, S., Jansen, B. J., Wong, K. F., & Cha, M. 2016. Detecting rumors from microblogs with recurrent neural networks. *IJCAI'16: Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, 3818–3824. Retrieved from <https://dl.acm.org/doi/10.5555/3061053.3061153>
- Ma, J., Gao, W., & Wong, K.-F. 2018a. Detect Rumor and Stance Jointly by Neural Multi-task Learning. *Companion of the The Web Conference 2018 on The Web Conference 2018 - WWW '18*, 585–593. <https://doi.org/10.1145/3184558.3188729>
- Ma, J., Gao, W., & Wong, K.-F. 2018b. Rumor Detection on Twitter with Tree-structured Recursive Neural Networks. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1980–1989. <https://doi.org/10.18653/v1/P18-1184>
- Mendoza, M., Poblete, B., & Castillo, C. 2010. Twitter under crisis. *Proceedings of the First Workshop on Social Media Analytics - SOMA '10*, 71–79. <https://doi.org/10.1145/1964858.1964869>
- Nguyen, T. N., Li, C., & Niederée, C. 2017. On Early-Stage Debunking Rumors on Twitter: Leveraging the Wisdom of Weak Learners. In *Lecture Notes in Computer Science: Vol. 10540 LNCS* (pp. 141–158). [https://doi.org/10.1007/978-3-319-67256-4\\_13](https://doi.org/10.1007/978-3-319-67256-4_13)
- Rath, B., Gao, W., Ma, J., & Srivastava, J. 2017. From Retweet to Believability: Utilizing Trust to Identify Rumor Spreaders on Twitter. *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*, 179–186. <https://doi.org/10.1145/3110025.3110121>
- Rosengren, E., Radev, D. R., Mei, Q., & Arbor, A. 2011. Rumor has it: identifying misinformation in microblogs. *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, 1589–1599. Retrieved from <https://dl.acm.org/citation.cfm?id=2145602>
- Vosoughi, S., Roy, D., & Aral, S. 2018. The spread of true and false news online. *Science*, 359(6380), 1146–1151. <https://doi.org/10.1126/science.aap9559>
- Wang, S., Moise, I., Helbing, D., & Terano, T. 2017. Early Signals of Trending Rumor Event in Streaming Social Media. *2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)*, 2, 654–659. <https://doi.org/10.1109/COMPSAC.2017.115>
- Wu, K., Yang, S., & Zhu, K. Q. 2015. False rumors detection on Sina Weibo by propagation structures. *2015 IEEE 31st International Conference on Data Engineering*, 2015-May, 651–662. <https://doi.org/10.1109/ICDE.2015.7113322>
- Wu, L., Li, J., Hu, X., & Liu, H. 2017. Gleaning Wisdom from the Past: Early Detection of Emerging Rumors in Social Media. In *Proceedings of the 2017 SIAM International Conference on Data Mining* (pp. 99–107). <https://doi.org/10.1137/1.9781611974973.12>
- Yang, F., Liu, Y., Yu, X., & Yang, M. 2012. Automatic detection of rumor on Sina Weibo. *Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics - MDS '12*, 2, 1–7. <https://doi.org/10.1145/2350190.2350203>
- Zhang, Y., Bian, K., Chen, L., Dong, S., Song, L., & Li, X. 2018. Early Detection of Rumors in Heterogeneous Mobile Social Network. *2018 IEEE Third International Conference on Data Science in*

- Cyberspace (DSC)*, 294–301.  
<https://doi.org/10.1109/DSC.2018.00049>
- Zhao, Z., Resnick, P., & Mei, Q. 2015. Enquiring Minds: Early Detection of Rumors in Social Media from Enquiry Posts. *Proceedings of the 24th International Conference on World Wide Web - WWW '15*, 1395–1405.  
<https://doi.org/10.1145/2736277.2741637>
- Zhou, K., Shu, C., Li, B., & Lau, J. H. 2019. Early Rumour Detection. *Proceedings of the 2019 Conference of the North*, 1614–1623.  
<https://doi.org/10.18653/v1/N19-1163>

- Zubiaga, A., Aker, A., Bontcheva, K., Liakata, M., & Procter, R. 2018. Detection and Resolution of Rumours in Social Media: A Survey. *ACM Computing Surveys*, 51(2), 1–36.  
<https://doi.org/10.1145/3161603>
- Zubiaga, A., Liakata, M., & Procter, R. 2016. *Learning Reporting Dynamics during Breaking News for Rumour Detection in Social Media*. *Computing Research Repository*, arXiv:1610.07363. Version 1
- Zubiaga, A., Liakata, M., & Procter, R. 2017. Exploiting Context for Rumour Detection in Social Media. In A. Jatowt, E.-P. Lim, Y. Ding, A. Miura, T. Tezuka, G. Dias, ... B. T. Dai (Eds.), *Lecture Notes in Computer Science: Vol. 10539 LNCS* (pp. 109–123).  
[https://doi.org/10.1007/978-3-319-67217-5\\_8](https://doi.org/10.1007/978-3-319-67217-5_8)

# Graph Attention Network with Memory Fusion for Aspect-level Sentiment Analysis

Li Yuan<sup>†</sup>, Jin Wang<sup>†,1</sup>, Liang-Chih Yu<sup>‡,\*,2</sup> and Xuejie Zhang<sup>†,3</sup>

<sup>†</sup>School of Information Science and Engineering, Yunnan University, Yunnan, P.R. China

<sup>‡</sup>Department of Information Management, Yuan Ze University, Taiwan

\*Innovation Center for Big Data and Digital Convergence Yuan Ze University, Taiwan

Contact : {wangjin<sup>1</sup>, xjzhang<sup>3</sup>}@ynu.edu.cn, lcyu@saturn.yzu.edu.tw<sup>2</sup>

## Abstract

Aspect-level sentiment analysis(ASC) predicts each specific aspect term's sentiment polarity in a given text or review. Recent studies used attention-based methods that can effectively improve the performance of aspect-level sentiment analysis. These methods ignored the syntactic relationship between the aspect and its corresponding context words, leading the model to focus on syntactically unrelated words mistakenly. One proposed solution, the graph convolutional network (GCN), cannot completely avoid the problem. While it does incorporate useful information about syntax, it assigns equal weight to all the edges between connected words. It may still incorrectly associate unrelated words to the target aspect through the iterations of graph convolutional propagation. In this study, a graph attention network with memory fusion is proposed to extend GCN's idea by assigning different weights to edges. Syntactic constraints can be imposed to block the graph convolutional propagation of unrelated words. A convolutional layer and a memory fusion were applied to learn and exploit multiword relations and draw different weights of words to improve performance further. Experimental results on five datasets show that the proposed method yields better performance than existing methods. The code of this paper is available at <https://github.com/YuanLi95/GATT-For-Aspect>.

## 1 Introduction

Aspect-level sentiment classification is a fine-grained subtask in sentiment analysis (Wang et al., 2019; Peng et al., 2020). Given a sentence and an aspect that appears in the sentence, ASC aims to determine the sentiment polarity of that aspect (e.g., negative, neutral, or positive). For example, a review of a restaurant “*The price is reasonable although the service is poor.*” expresses a *positive* sentiment for the *price* aspect, but also conveys a

*negative* sentiment for the *service* aspect, as shown in Figure 1. Such a technique is widely used to analyze online posts reviews, mainly from Amazon reviews or Twitter, to help raise the ability to understand consumer needs or experiences with a product, guiding a manufacturer towards product improvement. Aspect-level sentiment classification is much more complicated than sentence-level sentiment classification. ASC task is necessary to identify the parts of the sentence that describe the correspondence between multiple aspects. Traditional methods mostly use shallow machine learning models with hand-crafted features to build sentiment classifiers for the ASC task (Jiang et al., 2011; Wagner et al., 2014). However, the process for manual feature engineering is time-consuming and labor-intensive as well as limited in classification performance

Recently, with the development of deep learning techniques, various attention-based neural models have achieved remarkable success in ASC. (Wang et al., 2016; Ma et al., 2017; Chen et al., 2017; Gu et al., 2018; Tang et al., 2019). However, these methods ignored the syntactic dependence between context words and aspects in a sentence. As a result, the current attention model may inappropriately focus on syntactically unrelated context words. As shown in Figure 1, when predicting the emotional polarity of *price*, the attention mechanism may focus on the word *poor*, which is not related to its syntax.

To address this issue, Zhang et al. (2019) built a graph convolutional network (GCN) over a dependency tree to exploit syntactical information and word dependencies. However, the model assigns equal weight to the edges connected between words so that words may mistakenly associate syntactically unrelated words to the target aspect through iterations of graph convolutional propagation. As indicated in Figure 1, after three iterations, both *reasonable* (yellow lines) and *poor* (red lines) may be

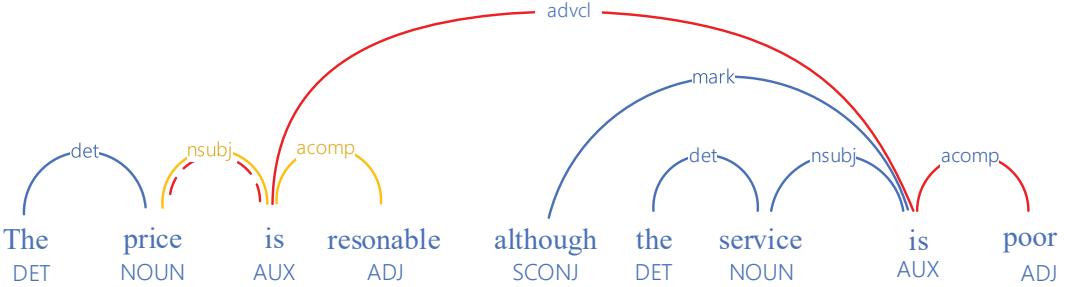


Figure 1: Grammatical Relational Examples.

identified as descriptors of the aspect *price*, which is incorrect. As a result, the model will falsely classify the aspect *price* as a negative sentiment.

In this paper, a graph attention model with memory fusion was proposed. This model extends the idea of graph convolutional networks in two aspects. First, the graph attention mechanism is applied to assign different weights to the edge, so the syntactical constraints can be imposed to block the propagation of syntactically unrelated words to the target aspect. Second, a convolutional operation is applied to extract local information to exploit multi-word relations, such as *not good* and *far from perfect*, which can further improve the performance. To integrate all features, a memory fusion layer, which is similar to a memory network, is applied to draw different weights for words according to their contribution to the final classification. Experiments are conducted on five datasets demonstrate how the proposed model outperforms baselines for aspect-level sentiment analysis.

The remainder of this paper is organized as follows. Section 2 briefly reviews the existing works for aspect-level sentiment analysis. Section 3 presents a detailed description of the proposed graph attention model with memory fusion. Section 4 summarizes the implementation details and experimental results. The conclusions of this study are finally drawn in Section 5.

## 2 Related Works

Aspect-level sentiment classification is an important branch of sentiment classification, aiming to identify the sentiment polarity of an aspect target in a sentence. ASC methods can be divided into traditional and deep learning methods. Traditional methods usually used feature-based machine learning algorithms, such as a feature-based support

vector machine (SVM) (Kiritchenko et al., 2014). Due to the inefficiency of manually constructed features, several neural network methods have been proposed for aspect-level sentiment analysis (Jiang et al., 2011), which are mainly based on long short-term memory (LSTM) (Tang et al., 2016a; Wang et al., 2020). Tang et al. (2016b) indicated that the ASC task’s challenge is to identify better the semantic correlation between context words and aspect words so that several recent works widely applied an attention mechanism and achieved good performance. Ma et al. (2017) used an interactive attention network to obtain a two-way attention representation of context words and aspect words. Huang et al. (2018) proposed a joint model based on an attention mechanism to model aspects and sentences. Tang et al. (2019) proposed a self-supervised attention model that can dynamically update attention weights.

Yao et al. (2019) introduced the graph convolutional network into the sentiment classification task and achieved good performance. Subsequently, Zhang et al. (2019) proposed to use GCN on the dependency tree of a sentence to exploit the long-range syntactic information for the ASC task.

## 3 Graph Attention Network with Memory Fusion

The proposed graph attention network with memory fusion is mainly composed of the following four parts: a context encoder, a graph attention layer, a convolutional layer and a memory fusion layer, as shown in Figure 2 . The context encoder employs a vanilla bidirectional LSTM to capture the textual features. It contains a word embedding layer and a BiLSTM layer to produce a hidden representation of the text. Taking the hidden representation as input, the graph attention layer (G-ATT)

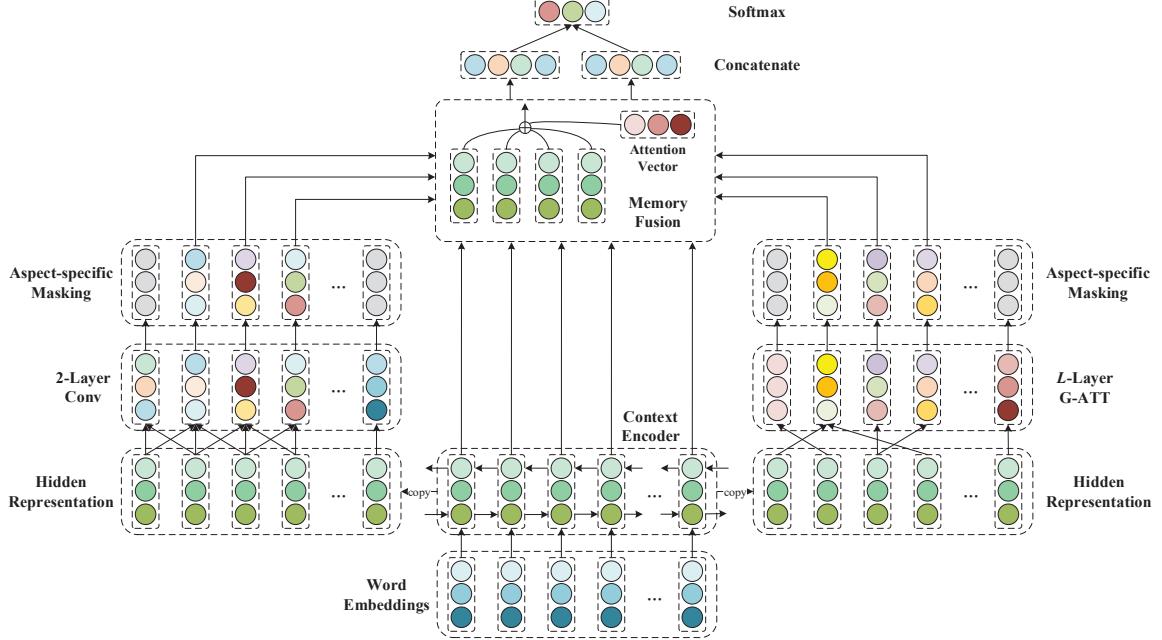


Figure 2: The overall architecture of the proposed graph attention network with memory fusion.

is trained on the dependency tree to mine explicit structural information between words. The convolutional layer was used to extract the local information around the sentiment word, which can dynamically deal with non-single word aspects such as *not good* and *far from perfect*, instead of only taking the average of its vectors. To merge all features, we adopt a memory fusion layer similar to a memory network (Tang et al., 2016b), which can assign different weights to the context words according to their contribution to the final classification. The detailed description is presented as follows.

### 3.1 Context Encoder

Given a sentence  $\mathbf{x} = [x_1, x_2, \dots, x_{\tau+1}, \dots, x_{\tau+m}, \dots, x_n]$  containing  $n$  words, the target aspect starts from the  $(\tau + 1)$ -th word with a length of  $m$ . A BiLSTM was applied as context encoder, which can capture long-distance dependencies within the sentence. We average the hidden representation of both the forward direction and backward direction to obtain the contextual representation, defined as,

$$(\vec{h}_i^E, \vec{c}_i^E) = LSTM(x_i, \vec{h}_{i-1}^E, \vec{c}_{i-1}^E) \quad (1)$$

$$(\overleftarrow{h}_i^E, \overleftarrow{c}_i^E) = LSTM(x_i, \overleftarrow{h}_{i+1}^E, \overleftarrow{c}_{i+1}^E) \quad (2)$$

$$h_i = (\vec{h}_i^E \oplus \overleftarrow{h}_i^E)/2 \quad (3)$$

where  $\oplus$  is an element-wise addition operator;  $\vec{h}_i^E \in \mathbb{R}^{d_h}$ ,  $\overleftarrow{h}_i^E \in \mathbb{R}^{d_h}$  and  $h_i \in \mathbb{R}^{d_h}$  are

the forward, backward and output representation, respectively; and  $d_h$  is the dimension of hidden state. Thus, the final representation of the context encoder can be denoted as  $H^E = [h_1^E, h_2^E, \dots, h_{\tau+1}^E, \dots, h_{\tau+m}^E, \dots, h_{\tau+m}^E]$ .

### 3.2 Graph Attention Layer

The graph attention (G-ATT) layer learns syntactically relevant words to the target aspect on the dependency tree<sup>1</sup>, which is widely used in several NLP tasks to effectively identify the relationships and roles of words. After parsing the given sentence as a dependency tree, the adjacency matrix was built from the tree topology. It is worth noting that the dependency tree is a directed graph. Therefore, the graph attention mechanism was applied with consideration of the direction, but the mechanism could be adapted to the undirectional-aware scenario. Therefore, we propose a variant on dependency graphs that are undirectional. The obtained hidden state  $H^E \in \mathbb{R}^{n \times d_h}$  was fed into a stacked G-ATT model, which was performed in a multilayer fashion with an  $L$  graph attention layer.

In practice, the representation in the  $l$ -the layer was not immediately fed into the G-ATT layer. To enhance the relevance of the context words to the corresponding aspect, we adopted a position weight function to the representation of word  $i$  in layer  $l$ ,

<sup>1</sup>We use spaCy toolkit: <https://spacy.io/>.

which is widely used in previous works (Li et al., 2018; Zhang et al., 2019), defined as,

$$q_i = \begin{cases} 1 - \frac{\tau+1-i}{n} & 1 \leq i < \tau + 1 \\ 0 & \tau + 1 \leq i \leq \tau + m \\ 1 - \frac{i-\tau-m}{n} & \tau + m < i \leq n \end{cases} \quad (4)$$

$$\hat{h}_i^l = q_i h_i^l \quad (5)$$

where  $q_i \in \mathbb{R}$  is the position weight to word  $i$ .

In each layer, an attention coefficient  $\alpha_{ij}^l$  was applied to measure the importance between word  $i$  and word  $j$ , defined as,

$$\alpha_{i,j}^l = \frac{\exp\left(\text{LeakyReLU}(\mathbf{a}^T [\mathbf{W}_\alpha^l \hat{h}_i^l || \mathbf{W}_\alpha^l \hat{h}_j^l])\right)}{\sum_{k \in \mathcal{N}_i} \exp\left(\text{LeakyReLU}(\mathbf{a}^T [\mathbf{W}_\alpha^l \hat{h}_i^l || \mathbf{W}_\alpha^l \hat{h}_k^l])\right)} \quad (6)$$

where  $\mathcal{N}_i$  is the set of the neighbor of word  $i$  and  $\mathbf{W}_\alpha^l \in \mathbb{R}^{d_h \times d_h}$  is a shared weight matrix applied to perform linear transformation to each word in order to obtain sufficient express ability of high-level representation.  $||$  is the concatenation operator,  $\mathbf{a} \in \mathbb{R}^{2d_h}$  is a weight vector, and the leaky rectified linear unit (LeakyReLU) is the non-linearity.

To stabilize the learning process of the graph's attention, we implement  $K$  different attention with the same parameter settings, which is similar to the multi-head attention mechanism proposed by Vaswani et al. (2017). Thus, the final representation  $h_i^{l+1}$  of word  $i$  in layer  $l+1$  can be obtained as,

$$h_i^{l+1} = \text{ReLU}\left(\frac{1}{K} \sum_{k=1}^K \sum_{j \in \mathcal{N}_i} \alpha_{i,j}^{l,k} \mathbf{W}_k^l \hat{h}_j^l\right) \quad (7)$$

where  $\alpha_{i,j}^{l,k}$  is the  $k$ -th attention coefficients computed by Eq. (6),  $\mathbf{W}_k^l$  is the corresponding weight matrix of  $k$ -th attention in  $l$ -th GAT layer, and the nonlinear function is ReLU. The final representation of the  $L$ -layer G-ATT is denoted as  $H^L = [h_1^L, h_2^L, \dots, h_{\tau+1}^L, \dots, h_{\tau+m}^L, \dots, h_n^L]$ ,  $h_i^L \in \mathbb{R}^{d_h}$ .

### 3.3 Convolutional Layer

The convolutional layer (Conv) was applied to extract local  $n$ -gram information which are composed of multiple sentiment words (e.g, *not good* and *far from perfect*), in order to improve the learning ability of the  $n$ -gram features. The hidden representation of context encoder  $H^E$  is fed into two convolutional layers. In each layer, we use  $F$  convolution filters to learn local  $n$ -gram features. In a

window of  $\omega$  words  $h_{i:i+\omega-1}$ , the filter  $f$ -th generates the feature map  $c_i^f$  as follows,

$$c_i^f = \text{ReLU}(\mathbf{W}^f \circ h_{i:i+\omega-1}^E + b^f) \quad (8)$$

where  $\circ$  is a convolutional operator,  $\mathbf{W}^f \in \mathbb{R}^{\omega \times d_h}$  and  $b^f \in \mathbb{R}^{d_h}$  respectively denote the weight matrix and bias,  $\omega$  is the length of the filter, and the non-linearity is ReLU. By concatenating all feature maps, the representation for word  $i$  will be  $h_i^c = [c_i^1, c_i^2, \dots, c_i^f, \dots, c_i^F]$ . To ensure that the shape of the output is consistent with the shape of the input in the convolutional layer, we set  $F$  to  $d_h$  and pad each sentence with zero vectors to the maximum input length in the corpora. Then, we send the feature maps to the second convolutional layer, which has a similar structure, to obtain the final representation of convolutional layer  $H^C = [h_1^C, h_2^C, \dots, h_{\tau+1}^C, \dots, h_{\tau+m}^C, \dots, h_n^C]$ ,  $h_i^C \in \mathbb{R}^{d_h}$ .

### 3.4 Aspect-Specific Masking

The aspect-specific masking layer aims to learn aspect-specific content for memory fusion and the final classification. Therefore, we mask out the hidden state vectors of the input from the G-ATT and Conv layer, i.e.,  $H^L$  and  $H^C$ . Formally, we set all the vectors of non-aspect words to zero and leave the vectors of the aspect words unchanged, defined as,

$$h_i = \begin{cases} 0 & 1 \leq i < \tau + 1, \tau + m < i \leq n \\ h_i & \tau + 1 \leq i \leq \tau + m \end{cases} \quad (9)$$

The output vector of the G-ATT layer after the mask operation is  $H_{\text{masked}}^L = [0, \dots, h_{\tau+1}^L, \dots, h_{\tau+m}^L, \dots, 0]$ , which has perceived contexts around the aspect so both syntactical dependencies and the long-range multiword relations can be considered. Similarly, the output representation of the convolutional layer after the mask operation is  $H_{\text{mask}}^C = [0, \dots, h_{\tau+1}^C, \dots, h_{\tau+m}^C, \dots, 0]$ .

### 3.5 Memory Fusion

Memory fusion aims to learn the final representation related to the meaning of aspect words. The idea is to retrieve significant features that are semantically relevant to the aspect words from the hidden representation by aligning the vectors of both G-ATT and Conv to the hidden vectors. Formally, we calculate the attention score for the  $i$ -th word in  $H^E$  and  $j$ -th word in  $H^L$ , defined as,

Dataset		Positive	Neutral	Negative	Total	Max Length	Mean Length
<b>Twitter</b>	Train	1561	3127	1560	6248	43	19
	Test	173	346	173	692	39	19
<b>Lap14</b>	Train	994	464	870	2328	81	21
	Test	341	169	128	638	70	17
<b>Rest14</b>	Train	2164	637	807	3608	77	18
	Test	728	196	196	1120	68	17
<b>Rest15</b>	Train	912	36	256	1204	72	15
	Test	326	34	182	542	61	17
<b>Rest16</b>	Train	1240	69	439	1748	72	16
	Test	469	30	117	616	77	18

Table 1: The summary of datasets

$$e_i = \sum_{j=1}^n h_i^{L^T} \mathbf{W}_l h_j^E = \sum_{j=\tau+1}^{\tau+m} h_i^{L^T} \mathbf{W}_l h_j^E \quad (10)$$

where  $\mathbf{W}_l \in \mathbb{R}^{d_h \times d_h}$  is a bilinear term that interacts with these two vectors and captures the specific semantic relations. According to Socher et al. (2013), such a tensor operator can be used to model complicated compositions between those vectors. Therefore, the attention score weight and final representation of G-ATT are computed as,

$$\alpha_i = \frac{\exp(e_i)}{\sum_{k=1}^n \exp(e_k)} \quad (11)$$

$$\mathbf{s}_g = \sum_{i=1}^n \alpha_i h_i^E \quad (12)$$

Accordingly, the final representation of the Conv layer is computed as,

$$r_i = \sum_{j=1}^n h_i^{C^T} \mathbf{W}_c h_j^E = \sum_{j=\tau+1}^{\tau+m} h_i^{C^T} \mathbf{W}_c h_j^E \quad (13)$$

$$\beta_i = \frac{\exp(r_i)}{\sum_{k=1}^n \exp(r_k)} \quad (14)$$

$$\mathbf{s}_c = \sum_{i=1}^n \beta_i h_i^E \quad (15)$$

### 3.6 Sentiment Classification

After obtaining representation  $\mathbf{s}_g$  and  $\mathbf{s}_c$ , they are fed into a fully connected layer and then a *softmax* layer to generate a probability distribution over the classes,

$$\hat{y} = \text{softmax}(\mathbf{W}_s [\mathbf{s}_g || \mathbf{s}_c] + b_s) \quad (16)$$

where  $\mathbf{W}_s$  and  $b_s$  respectively denote the weights and bias in the output layer. Thus, given a training

set  $\{\mathbf{x}^{(t)}, y^{(t)}\}_{t=1}^T = 1$ , where  $\mathbf{x}^{(t)}$  is a training sample,  $y^{(t)}$  is the corresponding actual sentiment label, and  $T$  is the number of training samples in the corpus. The training goal is to minimize the cross-entropy  $\mathcal{L}_{cls}(\theta)$  defined as,

$$\mathcal{L}_{cls}(\theta) = -\frac{1}{T} \sum_{t=1}^T \log p(\hat{y}^{(t)} | \mathbf{x}^{(t)}; \theta) + \lambda \|\theta\|_2^2 \quad (17)$$

where  $\theta$  denotes all trainable parameters. To avoid overfitting, an  $L_2$ -regularization  $\lambda \|\theta\|_2^2$  is also introduced to the loss function in the training phase, where  $\lambda$  is the decay factor.

## 4 Experimental Results

This section conducts comparative experiments on five corpora against several previously proposed methods for aspect-level sentiment analysis. The experimental setting and empirical results are then presented in detail.

### 4.1 Dataset

To compare the proposed model with other aspect-level sentiment analysis models, we conduct experiments on the following five commonly used datasets: **Twitter** was originally proposed by Dong et al. (2014) and contains several Twitter posts, while the other four corpora (**Lap14**, **Rest14**, **Rest15**, **Rest16**) were respectively retrieved from SemEval 2014 task 4 (Pontiki et al., 2014), SemEval 2015 task 12 (Pontiki et al., 2015) and SemEval 2016 Task 5 (Pontiki et al., 2016), which include two types of data, i.e., reviews of laptops and restaurants. The statistical descriptions of these corpora are shown in Table 1. We use accuracy and Macro-average  $F_1$ -score as evaluation metrics; these are commonly used in ASC task (Huang and Carley, 2019; Zhang et al., 2019). A higher accuracy or  $F_1$ -score indicates better prediction performance

Model	Twitter		Lap14		Rest14		Rest15		Rest16	
	Acc	F1								
<b>LSTM</b>	69.56	67.70	69.29	63.09	78.13	67.47	77.37	55.17	86.80	63.88
<b>TD-LSTM</b>	70.81	69.11	70.45	64.78	79.47	69.01	78.23	57.25	87.17	64.89
<b>MemNet</b>	71.48	69.90	70.64	65.17	79.61	69.64	77.31	58.28	85.44	65.99
<b>IAN</b>	72.50	70.81	72.05	67.38	79.26	70.09	78.54	52.65	84.74	55.21
<b>RAM</b>	69.36	67.30	74.49	<b>71.35</b>	80.23	70.80	78.85	<b>61.97</b>	88.92	68.23
<b>AOA</b>	72.30	70.20	72.62	67.52	79.97	70.42	78.17	57.02	87.50	66.21
<b>TNet-LF</b>	<b>72.98</b>	<b>71.43</b>	74.61	70.14	80.42	71.03	78.47	59.47	<b>89.07</b>	<b>70.43</b>
<b>ASGCN</b>	72.15	70.40	<b>75.55</b>	71.05	<b>80.77</b>	<b>72.02</b>	<b>79.89</b>	61.89	88.99	67.48
<b>G-ATT-U</b>	73.60	<b>72.12</b>	<b>76.18</b>	<b>72.23</b>	<b>81.59</b>	<b>72.65</b>	<b>81.18</b>	<b>64.07</b>	<b>89.06</b>	71.97
<b>G-ATT-D</b>	<b>73.89</b>	71.82	75.75	71.52	80.89	71.68	80.93	64.03	88.81	<b>72.36</b>

Table 2: Model comparison results (%). In the case of random initialization, the average accuracy of the 3 runs and the macro  $F_1$ -score. The best results of its baseline model and our model are shown in bold.

Model	Twitter		Lap14		Rest14		Rest15		Rest16	
	Acc	F1								
<b>ASGCN-DG</b>	72.15	70.40	75.55	71.05	80.77	72.02	79.89	61.89	88.99	67.48
<b>G-ATT-U</b>	73.60	<b>72.12</b>	<b>76.18</b>	<b>72.23</b>	<b>81.59</b>	<b>72.65</b>	<b>81.18</b>	<b>64.07</b>	<b>89.06</b>	<b>71.97</b>
<b>G-ATT-U w/o Pos</b>	<b>73.74</b>	<b>72.00</b>	75.13	71.26	<b>81.82</b>	<b>73.91</b>	80.07	62.42	<b>88.69</b>	69.54
<b>G-ATT-U w/o Mask</b>	73.36	71.47	<b>75.24</b>	70.70	80.15	70.49	79.89	<b>62.78</b>	88.53	<b>70.34</b>
<b>G-ATT-U w/o GAT</b>	73.03	71.04	74.56	71.23	80.21	71.16	80.38	61.31	87.66	68.27
<b>G-ATT-U w/o Conv</b>	73.23	71.22	74.82	<b>71.35</b>	80.86	71.77	<b>80.54</b>	62.02	87.39	69.22

Table 3: Ablation study results (%). Accuracy and macro F1-scores are the average value over 3 runs with random initialization.

## 4.2 Implementation Details

To comprehensively evaluate the proposed model, we selected the following baseline methods, which are introduced as follows:

- **LSTM** (Tang et al., 2016a) uses the standard LSTM model to send the state of the last layer to the *softmax* layer to obtain the output of sentiment probability.
- **TD-LSTM** (Tang et al., 2016a) connects aspect word embedding and context word embedding to obtain the final word embedding representation, and the two sides of the aspect word are respectively modeled by LSTM to obtain the hidden layer representation.
- **MemNet** (Tang et al., 2016b) consists of a multilevel memory network, which effectively retains context and aspect information.
- **IAN** (Ma et al., 2017) exchanges information between context and aspect as an interactive attention model.
- **RAM** (Chen et al., 2017) learns sentence representation by layers consisting of an attention-based aggregation of word features and a GRU cell with multilayer architecture.

- **AOA** (Huang et al., 2018) captures the interaction between context and aspect words by jointly modeling aspects and sentences.
- **TNet-LFT** (Li et al., 2018) increases the retention of context information through a context retention conversion mechanism.
- **ASGCN** (Zhang et al., 2019) uses external grammatical information through the graph convolution neural network, while aspect obtains syntax-related context information.
- **G-ATT** uses either undirectional (**G-ATT-U**) or directional (**G-ATT-D**) graphs to represent the parsed tree-structure as the proposed model.

For all the models, the 300-dimensional GloVe vector (Pennington et al., 2014) pretrained on 840B Common Crawl was used as the initial word embedding. Words that do not appear in GloVe were initialized with a uniform distribution of  $U(-0.25, 0.25)$ . The hidden layer vectors’ dimensions are all 300, and all model weights are initialized with the *Xavier* normalization (Glorot and Bengio, 2010). RMSprop was used as the optimizer with a learning rate of 0.001 to train all the models. We set the  $L_2$ -regularization decay factor to 1e-4 and the batch size to 40. The negative input slope of LeakyReLU in the G-ATT layer is set to 0.2. All aforementioned

Aspect	Model	Attention Visualization	Prediction	Label
OS	ASGCN		neutral	positive
	Conv		positive	positive
	G-ATT			
Cajun shrimp	ASGCN		negative	positive
	Conv		positive	positive
	G-ATT			
Place	ASGCN		neutral	negative
	Conv		negative	negative
	G-ATT			

Table 4: Visualization of the proposed model.

hyperparameters are selected using a grid-search strategy. The epoch was set depending on an early stop strategy. The training processing stops after five epochs if there is no improvement. The experimental results are obtained by averaging the results of three random initialization runs.

### 4.3 Comparative Results

Table 2 shows the comparative results of G-ATT-D and G-ATT-U against several baselines. As indicated, G-ATT-U outperformed all baseline models by using  $F_1$ -score as a criterion. In terms of accuracy, except results slightly lower than the TNet-LF model on **Rest16**, both G-ATT-D and G-ATT-U achieved better performance. The rational reason is that the proposed model can capture both syntactic and local information, thus improving performance.

In addition, the improvement of the  $F_1$ -score of the proposed model on **Rest15** and **Rest16** is huge compared to the baselines, which is 2.1% and 1.5%,

respectively. The possible reason is that the syntactical structure of the texts in **Rest15** and **Rest16** is more complicated than those in **Twitter**, **Lap14** and **Rest14**. The performance of directional version (G-ATT-D) is slightly higher than the undirectional version (G-ATT-U) on **Twitter**, **Rest15** and **Rest16**, while performance is slightly lower on **Lap14** and **Rest14**, indicating an undirectional syntax relationship that is more appropriate on those datasets.

### 4.4 Ablation Experiment

Table 3 shows the ablation experiments to investigate further how the models can benefit from each component. As indicated, removing the position weight (i.e., G-ATT-U w/o Pos) causes the performance on **Lap14**, **Rest15** and **Rest16** to decrease. However, the performance of G-ATT-U w/o Pos increases  $F_1$ -score by 1.26% when used on **Twitter** and **Rest14** since the local information is less important than syntactic. Removing the mask op-

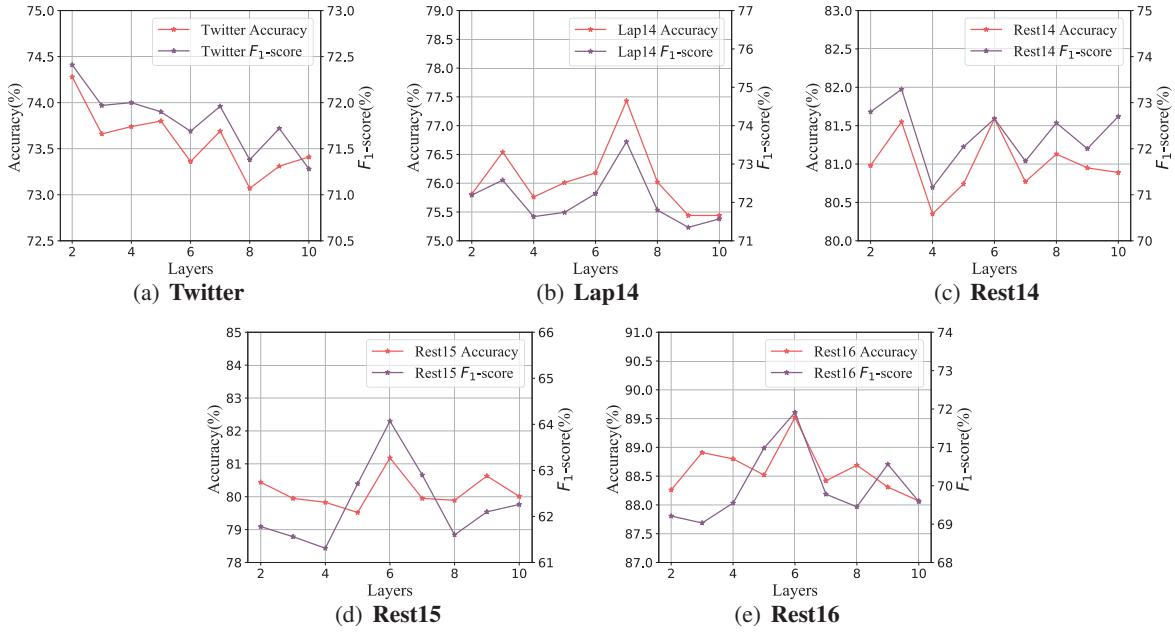


Figure 3: Effect of the number of G-ATT Layers

eration (i.e., G-ATT-U w/o mask) reduces the performance, which shows that the mask operation prevents the noise word from entering the final representation. Further, **Twitter**, **Lap14**, and **Rest14** are less syntactical, so the integration of position weight does not benefit or can even negatively benefit the results.

Besides, it is observed that G-ATT-U w/o Conv is generally better than G-ATT-U w/o G-ATT, which shows that the GAT layer benefits for the model are greater than the Conv layer, indicating that the contextual syntax-related information is more important than local information. Compared with ASGCN-DG, the proposed G-ATT-U w/o Conv achieved better performance, especially on **Twitter** and **Rest16**, with  $F_1$ -score improvements of 0.64% and 1.74%, respectively. This result shows that G-ATT-U w/o Conv outperformed the ASGCN model in most cases, indicating that graph attention layers with different edge weights are more effective than graph convolution layers with equal edge weights.

#### 4.5 Visualization

Memory fusion can capture both syntax-related and local information with the attention mechanism. For visualization, we selected three examples from **Lap14** and **Rest16** that are significantly improved by the proposed G-ATT model against the ASGCN-DG model. We conducted a visualization experiment using a heat map to show the attention score offered by parameters  $\alpha$  and  $\beta$  in Eq.(11) and Eq.(14), respectively, as shown in Table 4. The

color density is the attention score of each token. A deeper color indicates that more weight is assigned to the token according to its contribution to the final classification. As indicated, ASGCN allows the syntactically unrelated words to be associated with the target aspect by assigning equal weight to the edge, such as *great* for OS, *good* for *Cajun shrimp* and *not inviting* for place. Conversely, G-ATT-U tends to block graph convolution propagation from unrelated words to the target aspect by assigned attention weights to the edges. The convolution operation can also exploit some explicit structure, such as *not great* and *not inviting*. Such phrases are expressive and task-specific, thus improve performance.

#### 4.6 Number of GAT layers

Since G-ATT involves  $L$  layers of graph attention, we investigate whether the number of layers can determine the proposed model’s performance. As indicated, the best performance can be achieved when  $L$  is 2 on **Twitter**, 7 on **Lap14**, 3 on **Rest14** and 6 on **Rest15** and **Rest16**. When  $L$  is greater than 7, a decreasing trend in both metrics is presented. As  $L$  reaches 10, the model contains too many parameters and becomes more difficult to train.

### 5 Conclusions

In this study, a graph attention network with memory fusion is proposed for aspect-level sentiment analysis. A graph attention layer was implemented

to capture a context word’s syntactic relationship to the target aspect by learning different weights for edges to block the propagation from unrelated words. Moreover, a convolutional layer and a memory fusion were used to learn the local information and draw different weights for context words. Experimental results show that the G-ATT model yields better performance than the existing methods for aspect-based sentiment analysis. Besides, ablation studies and case studies are provided to prove the effectiveness of the proposed model further. Future works will improve the graph attention layer and dynamic to learn the attention score, so the proposed model can better integrate syntax-related context information.

## Acknowledgments

This work was supported by the National Natural Science Foundation of China (NSFC) under Grant No. 61966038, 61702443 and 61762091, and in part by the Ministry of Science and Technology, Taiwan, ROC, under Grant No. MOST107-2628-E-155-002-MY3. The authors would like to thank the anonymous reviewers for their constructive comments.

## References

- Peng Chen, Zhongqian Sun, Lidong Bing, and Wei Yang. 2017. Recurrent attention network on memory for aspect sentiment analysis. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP-2017)*, pages 452–461.
- Li Dong, Furu Wei, Chuanqi Tan, Duyu Tang, Ming Zhou, and Ke Xu. 2014. Adaptive Recursive Neural Network for Target-dependent Twitter Sentiment Classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL-2014)*, pages 49–54.
- Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 249–256.
- Shuqin Gu, Lipeng Zhang, Yuxian Hou, and Yin Song. 2018. A position-aware bidirectional attention network for aspect-level sentiment analysis. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING-2018)*, pages 774–784.
- Binxuan Huang and Kathleen M Carley. 2019. Parameterized convolutional neural network for aspect-level sentiment classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP-2018)*, pages 1091–1096.
- Binxuan Huang, Yanglan Ou, and Kathleen M. Carley. 2018. Aspect level sentiment classification with attention-over-attention neural networks. In *Proceedings of the International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation*, pages 197–206.
- Long Jiang, Mo Yu, Ming Zhou, Xiaohua Liu, and Tiejun Zhao. 2011. Target-dependent twitter sentiment classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL-2011)*, pages 151–160.
- Svetlana Kiritchenko, Xiaodan Zhu, Colin Cherry, and Saif M. Mohammad. 2014. NRC-Canada-2014: Detecting aspects and sentiment in customer reviews. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval-2014)*, pages 437–442.
- Xin Li, Lidong Bing, Wai Lam, and Bei Shi. 2018. Transformation networks for target-oriented sentiment classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL-2018)*, pages 946–956.
- Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. 2017. Interactive attention networks for aspect-level sentiment classification. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI-2017)*, pages 4068–4074.
- Bo Peng, Jin Wang, and Xuejie Zhang. 2020. Adversarial learning of sentiment word representations for sentiment analysis. *Information Sciences*, 541:426–441.
- Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP-2014)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.
- Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammad AL-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, Véronique Hoste, Marianna Apidianaki, Xavier Tannier, Natalia Loukachevitch, Evgeniy Kotelnikov, Nuria Bel, Salud María Jiménez-Zafra, and Gülsen Eryiğit. 2016. SemEval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 19–30, San Diego, California. Association for Computational Linguistics.

- Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. 2015. Semeval-2015 task 12: aspect based sentiment analysis. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval-2015)*, pages 486–495.
- Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 task 4: aspect-based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation, Proceedings (SemEval-2014)*, pages 27–35.
- Richard Socher, Alex Perelygin, and Jy Wu. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP-2013)*, pages 1631–1642.
- Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2016a. Effective LSTMs for target-dependent sentiment classification. In *Proceedings of the 26th International Conference on Computational Linguistics (COLING-2016)*, pages 3298–3307.
- Duyu Tang, Bing Qin, and Ting Liu. 2016b. Aspect level sentiment classification with deep memory network. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-2016)*, pages 214–224.
- Jialong Tang, Ziyao Lu, Jinsong Su, Yubin Ge, and Linfeng Song. 2019. Progressive self-supervised attention learning for aspect-level sentiment analysis. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL-2019)*, pages 557–566. Association for Computational Linguistics.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems (NIPS-2017)*, pages 5999–6009.
- Joachim Wagner, Piyush Arora, Santiago Cortes, Utsab Barman, Dasha Bogdanova, Jennifer Foster, and Lamia Tounsi. 2014. DCU: Aspect-based polarity classification for semeval task 4. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval-2014)*, pages 223–229.
- Jin Wang, Liang-chih Yu, K. Robert Lai, and Xuejie Zhang. 2016. Community-Based Weighted Graph Model for Valence-Arousal Prediction of Affective Words. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(11):1957–1968.
- Jin Wang, Liang-Chih Yu, K. Robert Lai, and Xuejie Zhang. 2019. Investigating dynamic routing in tree-structured LSTM for sentiment analysis. In *Proceedings of the 2019 Conference on Empirical Methods* *in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3432–3437. Association for Computational Linguistics.
- Jin Wang, Liang Chih Yu, K. Robert Lai, and Xuejie Zhang. 2020. Tree-structured regional CNN-LSTM model for dimensional sentiment analysis. *IEEE/ACM Transactions on Audio Speech and Language Processing*, 28:581–591.
- Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI-2019)*, pages 7370–7377.
- Chen Zhang, Qiuchi Li, and Dawei Song. 2019. Aspect-based sentiment classification with aspect-specific graph convolutional networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP-2019)*, pages 4568–4578.

# FERNet: Fine-grained Extraction and Reasoning Network for Emotion Recognition in Dialogues

Yingmei Guo, Zhiyong Wu, Mingxing Xu

Department of Computer Science and Technology

Beijing National Research Center for Information Science and Technology

Tsinghua University, Beijing, China

guoym18@mails.tsinghua.edu.cn, zywu@sz.tsinghua.edu.cn

xumx@tsinghua.edu.cn

## Abstract

Unlike non-conversation scenes, emotion recognition in dialogues (ERD) poses more complicated challenges due to its interactive nature and intricate contextual information. All present methods model historical utterances without considering the content of the target utterance. However, different parts of a historical utterance may contribute differently to emotion inference of different target utterances. Therefore we propose Fine-grained Extraction and Reasoning Network (FERNet) to generate target-specific historical utterance representations. The reasoning module effectively handles both local and global sequential dependencies to reason over context, and updates target utterance representations to more informed vectors. Experiments on two benchmarks show that our method achieves competitive performance compared with previous methods.

## 1 Introduction

With the development of human-machine interaction (HMI) applications, textual dialogue scenes appear more frequently. These scenes request effective and high-performance emotion recognition systems helping in building empathetic machines (Young et al., 2018). Therefore, emotion recognition in dialogues (ERD) is getting growing attention from both academic and business community.

Different from non-conversation scenes, the ERD task poses a more complicated challenge of modeling context-sensitive dependencies. Most of existing approaches adopt Convolution Neural Network (CNN) (Krizhevsky et al., 2012), followed by a max-pooling layer to obtain utterance representations (Kim, 2014; Torres, 2018; Hazarika et al., 2018a,b; Majumder et al., 2019; Ghosal et al., 2019). The process proceeds without the guidance of the target utterance, thus generated historical

utterance representations are indistinguishable toward different target utterances. Emotion recognition may fail in cases where historical utterances express various emotions toward various targets, which may confuse the emotion recognition of target utterances. As Figure 1 shows, for different target utterances  $B_1$  and  $B_2$ , the model should attend the words “good service” and “bad food” in  $A_1$ , separately. In a word, it is desired to pay different attention to different words of a certain historical utterance to generate the target-specific historical utterance representation.

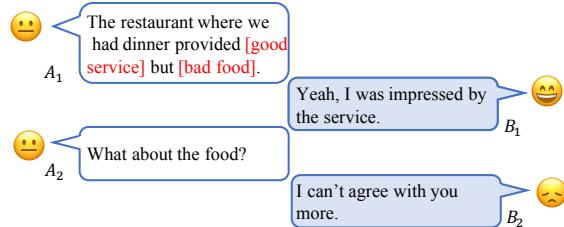


Figure 1: A dialogue shows that modeling intricate contextual information is crucial for emotion recognition.

In this paper, we propose Fine-grained Extraction and Reasoning Network (FERNet) to generate target-specific historical utterance representations conditioned on the content of target utterances by using the multi-head attention mechanism (Vaswani et al., 2017), extracting more fine-grained, relevant and contributing information for emotion recognition. Besides, we devise the reasoning module, which employs historical utterances as a sequence of triggers, and updates the representation of the target utterance to a more informed vector as it observes historical utterances through time. In the reasoning process, the module models both short-term and long-term sequential dependencies effectively. We demonstrate the effectiveness of our method on two benchmarks. Experimental results show that our method achieves competitive performance compared with previous methods.

## 2 Related Work

Primitive approaches deal with the ERD task as simple solely-sentence emotion recognition task with no consideration of the historical information (Joulin et al., 2016; Chen et al., 2016; Yang et al., 2016; Chatterjee et al., 2019).

To exploit contextual information, Poria et al. (2017); Huang et al. (2019); Jiao et al. (2019); Hazarika et al. (2018a,b); Torres (2018) use RNN architecture, Hazarika et al. (2018b,a) use conversational memory networks (Sukhbaatar et al., 2015), Torres (2018); Jiao et al. (2019) use attention mechanism and Ghosal et al. (2019) uses graph neural network.

Besides, Majumder et al. (2019); Hazarika et al. (2018a) propose to keep track of states of individual speakers throughout the dialogue and Ghosal et al. (2019) incorporates speaker information into edge types.

Some of these works consider the context following the target utterance such as Luo et al. (2018); Saxena et al. (2018); Ghosal et al. (2019) and some variants of Majumder et al. (2019). However, this condition is quite incompatible with some practical situations like real-time dialogue systems in which we possess no future utterances while handling the target utterance. So in our paper, we only focus on the setting that only historical utterances can be utilized.

## 3 Proposed Model

Each dialogue  $D$  consists of two parts denoted as  $D = \{(U, S)\}$ , where  $U = [u_1, u_2, \dots, u_n]$  is a sequence of utterances ordered based on their temporal occurrence.  $S = [s_1, s_2, \dots, s_n]$  denotes corresponding speakers and  $n$  is the number of utterances in the dialogue. The ERD task aims to predict  $Y = [y_1, y_2, \dots, y_n]$ , where  $y_i \in C$  ( $1 \leq i \leq n$ ) denotes the underlying emotion of the utterance  $u_i$ .  $C$  is the set of candidate emotion categories. The FERNNet consists of four successive modules: feature extraction module, attention module, reasoning module and output module. Figure 2 presents the overall architecture of the proposed model.

### 3.1 Feature Extraction Module

We use two multi-layer bidirectional Gated Recurrent Unit (bi-GRU) Networks (Tang et al., 2015) to accumulate contextual information from two directions for each word of target utterances and historical utterances, separately. The inputs consist of

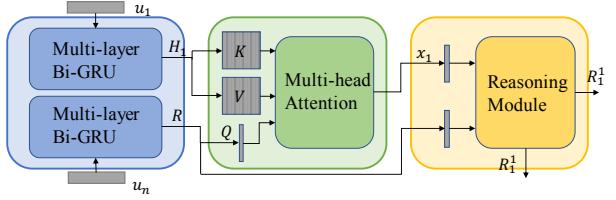


Figure 2: The overall architecture of the model.

300 dimensional pre-trained GloVe vectors (Pennington et al., 2014). The  $k$ -th contextual word representation  $h_k^l = [\vec{h}_k^l; \vec{h}_k^l]$  is generated by concatenating the hidden states of the  $k$ -th time steps of forward and backward GRU, where  $l$  is the number of layers.

### 3.2 Attention Module

We utilize multi-head attention mechanism (Vaswani et al., 2017) to focus on more relevant parts of each historical utterance according to the target utterance. We also employ residual connection (He et al., 2016) followed by layer normalization (Ba et al., 2016) to make model training easier.

The target-specific representations of historical utterances are obtained by:

$$X = \text{Concat}(\text{head}_1, \dots, \text{head}_t)W^O \quad (1)$$

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \quad (2)$$

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (3)$$

where queries  $Q = R$  are representations of target utterances, keys  $K$  and values  $V$  are contextual word representations for words of historical utterances.  $W_i^Q \in \mathbb{R}^{d \times d_k}, W_i^K \in \mathbb{R}^{d \times d_k}, W_i^V \in \mathbb{R}^{d \times d_v}$  and  $W^O \in \mathbb{R}^{td_v \times d}$  are parameter matrices, where  $d_k$  is the dimension of queries and keys,  $d_v$  is the dimension of values,  $d$  is the dimension of the output of feature extraction module and  $t$  is the number of heads.  $X = [x_1, x_2, \dots, x_{n-1}]$  are target-specific representations of historical utterances.

### 3.3 Reasoning Module

The reasoning module takes target-specific historical utterance representations  $[x_1, x_2, \dots, x_{n-1}]$  and target utterance representations  $R$  as inputs. Target utterance representations are updated through time and layers.

Each unit in this module takes two inputs:  $R$  and  $x_i$  ( $1 \leq i \leq n-1$ ). The  $t$ -th unit updates  $R$  according to  $x_t$  by:

$$z_t = \alpha(x_t, R) = \sigma(W^z(x_t \circ R) + b^z) \quad (4)$$

$$r_t = \beta(x_t, R) = \sigma(W^r(x_t \circ R) + b^r) \quad (5)$$

$$\tilde{R}_t = \rho(x_t, R) = \tanh(W^h[x_t; R] + b^h) \quad (6)$$

$$R_t = z_t r_t \tilde{R}_t + (1 - z_t) R_{t-1} \quad (7)$$

where  $z_t$  is the update gate,  $r_t$  is a reset function,  $\tilde{R}_t$  is the candidate of updated representation of target utterance and  $R_t$  is the updated representation after observing the  $t$ -th historical utterance.  $\sigma$  is sigmoid activation,  $\tanh$  is hyperbolic tangent activation,  $\circ$  is element-wise vector multiplication, and  $[;]$  is vector concatenation along the last dimension.  $W^z \in \mathbb{R}^{d \times d}$ ,  $W^r \in \mathbb{R}^{d \times d}$ ,  $W^h \in \mathbb{R}^{d \times 2d}$  are weight matrices,  $b^z \in \mathbb{R}^d$ ,  $b^r \in \mathbb{R}^d$ ,  $b^h \in \mathbb{R}^d$  are bias terms.

Specifically,  $z_t$  measures the relevance between the target utterance representation and the  $t$ th historical utterance representation for fine-controlled gating. Compared with global attention computed over all historical utterances, the gate can be considered as local attention which models short-term sequential dependency.  $r_t$  is a reset function to determine how much previous information should be ignored by resetting the candidate of updated representation of target utterance.

As shown in Sukhbaatar et al. (2015), multi-hop can perform reasoning over multiple facts more effectively. So we stack several layers with outputs of the current layer used as inputs to the next layer. Besides, to model more abundant information, we compute  $\overrightarrow{R}_t^l$  and  $\overleftarrow{R}_t^l$  in both forward and backward directions and add them together to get  $R_t^l$  as the updated representation of the  $t$ -th unit in  $l$ -th layer:

$$R_t^l = \overrightarrow{R}_t^l + \overleftarrow{R}_t^l \quad (8)$$

Finally, we get the updated representation of target utterance  $R^{update} = R_{n-1}^L$ , where  $n-1$  and  $L$  are the number of units and the number of layers in the reasoning module, respectively.

### 3.4 Output Module

After the feature extraction and reasoning modules, we obtain the updated representation of target utterance. To preserve original semantic content, we concatenate the updated representation and the original representation together:

$$R_{final} = [R^{update}; R] \quad (9)$$

We use a fully connected layer with softmax as activation to calculate emotion-class probabilities:

$$P = softmax(W^f R_{final} + b^f) \quad (10)$$

where  $W^f \in \mathbb{R}^{d_{class} \times 2d}$  is a weight matrix,  $b^f \in \mathbb{R}^{d_{class}}$  is a bias term and  $P \in \mathbb{R}^{d_{class}}$  are emotion-class probabilities.

## 4 Experiment

**Datasets** We perform experiments on two benchmarks: IEMOCAP (Busso et al., 2008) and AVEC (Schuller et al., 2012). They are multimodal datasets involved in two-way dynamic conversations. In this paper, we only focus on using textual modality to recognize the emotion. The data distribution is shown in Appendices.

**Evaluation Metrics** We use accuracy (Acc.), F1-score (F1) and weighted average F1-score (Average) as evaluation metrics for IEMOCAP dataset. Mean Absolute Error (MAE) and Pearson correlation coefficient (r) are used as metrics for AVEC dataset.

**Baselines** We compare the FERNet with following existing approaches: CNN (Kim, 2014), c-LSTM (Poria et al., 2017), c-LSTM+Attention (Poria et al., 2017), Memnet (Ba et al., 2016), CMN (Hazarika et al., 2018b), DialogueRNN (Majumder et al., 2019).

**Training Details** The training details such as hyper-parameters and settings we used are shown in Appendices.

### 4.1 Results

The overall results of experiments are shown in Table 1. We can see that our model outperforms baselines significantly on all evaluation metrics of both datasets. Specifically, our model surpasses DialogueRNN by 1.69% on weighted average F1-score. For AVEC dataset, our model lower mean absolute error by 0.03, 0.027, 0.009 and 0.31 for valence, arousal, expectancy and power, separately. We attribute the enhancement to the fundamental improvement of FERNet, which are generating target-specific representations of historical utterances and handling both short-term and long-term sequential dependencies.

### 4.2 Discussion and Analysis

**Parameters** We conduct experiments with different values of the number of historical utterances ( $N$ ) and the number of layers of reasoning module ( $L$ ) on the IEMOCAP dataset. Results are shown in Figure 4. We observe that as  $N$  increases, the performance of the model tends to be improved. This trend shows that adequate historical information

methods	IEMOCAP												AVEC									
	Happy		Sad		Neutral		Angry		Excited		Frustrated		Average		Valence	Arousal	Expectancy	Power				
	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1	MAE	r	MAE	r	MAE	r		
CNN	27.22	29.86	57.14	53.83	34.33	40.14	61.17	52.44	46.15	50.09	62.99	55.75	48.92	48.18	0.545	-0.01	0.542	0.01	0.605	-0.01	8.71	0.19
c-LSTM	29.17	34.43	57.14	60.87	54.17	51.81	57.06	56.73	51.17	57.95	67.19	58.92	55.21	54.95	0.194	0.14	0.212	0.23	0.201	0.25	8.90	-0.04
c-LSTM+Attention	30.56	35.63	56.73	62.90	57.55	53.00	59.41	59.24	52.84	58.85	65.88	59.41	56.32	56.19	0.189	0.16	0.213	0.25	0.190	0.24	8.67	0.10
Memnet	25.72	33.53	55.53	61.77	58.12	52.84	59.32	55.39	51.50	58.30	67.2	59.00	55.72	55.10	0.202	0.16	0.211	0.24	0.216	0.23	8.97	0.05
CMN	25.00	30.38	55.92	62.41	52.86	52.39	61.76	59.83	55.52	60.25	71.13	60.69	56.56	56.13	0.192	0.23	0.213	0.29	0.195	0.26	8.74	-0.02
DialogueRNN*	31.25	33.83	66.12	69.83	63.02	57.76	61.76	<b>62.50</b>	61.54	64.45	59.58	<b>59.46</b>	59.33	59.89	0.188	0.28	0.201	0.36	0.188	0.32	8.19	0.31
FERNNet	38.89	<b>40.14</b>	72.65	<b>70.22</b>	67.19	<b>61.50</b>	66.47	62.43	68.90	<b>68.21</b>	50.39	58.63	<b>61.80</b>	<b>61.58</b>	<b>0.158</b>	0.44	<b>0.174</b>	0.43	<b>0.179</b>	0.37	<b>7.88</b>	0.36

Table 1: Performance of FERNNet compared with baselines on the IEMOCAP dataset and AVEC dataset. Bold font denotes the best performances. \* presents the state-of-the-art method in the setting that only historical utterances can be utilized.

I'm getting married. [excited]  
 No way. [excited].  
 No way, when? When, when, when did it happen? [excited]  
 [1]Just a couple days ago. [excited]  
 I can't believe it.  
 I never thought you would get married. [excited]  
 [2]I know me neither. [excited]

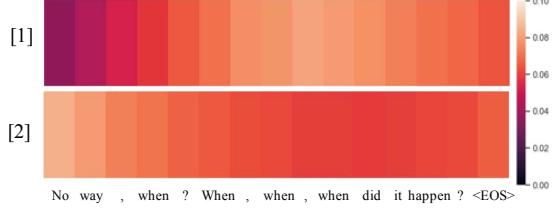


Figure 3: Average attention vectors across all attention heads for words of a historical utterance with regard to different target utterances. [1] shows the attention vector for the sentence "Just a couple days ago"; [2] shows the attention vector for the sentence "I know me either".

contributes to the performance of emotion recognition. However, a further increase of  $N$  degrades the performance of the model. It is mainly due to that there is too much-unrelated information confusing the model. As for  $L$ , the trend is similar to the parameter  $N$ . Models with hops in the range of 2-8 outperform the single layer variant. However, with  $L$  increasing, the reasoning module deepens and may cause the gradient vanishing problem which damages the performance of the model.

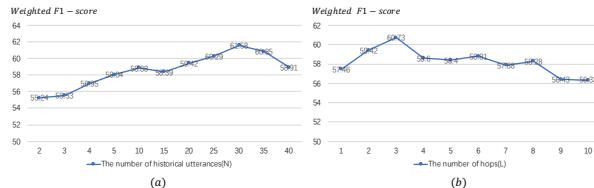


Figure 4: Performance of FERNNet with different values of  $N$  and  $L$ . In (a),  $L = 2$  and in (b),  $N = 20$ .

**Ablation Study** In order to demonstrate the effect of each module, we perform ablation studies. We compare the attention-based model with the attention-free model and replace the reasoning module with a memory network. As shown in Table 2, attention module and reasoning module both have a positive impact on model performance.

**Case Study and Error Analysis** We analyze the predicted results and find that misclassification often occurs when utterances are short. For example, our model classifies "what?" as "neutral", but the label is "excited". We think it is due to the lack of visual and audio modality. In this utterance,

methods	Acc.	F1
FERNNet without attention	58.84	58.58
FERNNet with memory network	59.77	59.33
<b>FERNNet</b>	<b>61.80</b>	<b>61.58</b>

Table 2: Performance of variants of FERNNet on the IEMOCAP dataset. Bold font denotes the best performances.

high pitched audio can provide vital information for recognizing the emotion. Besides, we find our model misclassifies several "excited" utterances as "happy" utterances, several "sad" utterances as "frustrated" utterances, and vice versa. The reason is that it is hard for the model to distinguish the subtle difference between these similar emotions.

Besides, we perform qualitative visualization of the attention module. The dialogue in Figure 3 shows that for different target utterances, the model allocates different attention to words of a historical utterance. It demonstrates the effectiveness of the attention module.

## 5 Conclusion

In this paper, we propose FERNNet to solve the ERD task. The model generates target-specific historical utterances according to the content of the target utterance using attention mechanism. The reasoning module effectively handles both local and global sequential dependencies to update the original representation of the target utterance to a more informed vector. Our model achieves competitive performance on two benchmarks.

## References

- Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.
- Carlos Busso, Murtaza Bulut, Chi-Chun Lee, Abe Kazemzadeh, Emily Mower, Samuel Kim, Jeanette N Chang, Sungbok Lee, and Shrikanth S Narayanan. 2008. Iemocap: Interactive emotional dyadic motion capture database. *Language resources and evaluation*, 42(4):335.
- Ankush Chatterjee, Umang Gupta, Manoj Kumar Chin-nakotla, Radhakrishnan Srikanth, Michel Galley, and Puneet Agrawal. 2019. Understanding emotions in text using deep learning and big data. *Computers in Human Behavior*, 93:309–317.
- Huimin Chen, Maosong Sun, Cunchao Tu, Yankai Lin, and Zhiyuan Liu. 2016. Neural sentiment classification with user and product attention. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1650–1659.
- Deepanway Ghosal, Navonil Majumder, Soujanya Poria, Niyati Chhaya, and Alexander Gelbukh. 2019. Dialoguegcn: A graph convolutional neural network for emotion recognition in conversation. *arXiv preprint arXiv:1908.11540*.
- Devamanyu Hazarika, Soujanya Poria, Rada Mihalcea, Erik Cambria, and Roger Zimmermann. 2018a. Icon: Interactive conversational memory network for multimodal emotion detection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2594–2604.
- Devamanyu Hazarika, Soujanya Poria, Amir Zadeh, Erik Cambria, Louis-Philippe Morency, and Roger Zimmermann. 2018b. Conversational memory network for emotion recognition in dyadic dialogue videos. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2122–2132.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.
- Chenyang Huang, Amine Trabelsi, and Osmar R Zaiane. 2019. Ana at semeval-2019 task 3: Contextual emotion detection in conversations through hierarchical lstms and bert. *arXiv preprint arXiv:1904.00132*.
- Wenxiang Jiao, Haiqin Yang, Irwin King, and Michael R Lyu. 2019. Higru: Hierarchical gated recurrent units for utterance-level emotion recognition. *arXiv preprint arXiv:1904.04446*.
- Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.
- Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.
- Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.
- Linkai Luo, Haiqing Yang, and Francis YL Chin. 2018. Emotionx-dlc: self-attentive bilstm for detecting sequential emotions in dialogue. *arXiv preprint arXiv:1806.07039*.
- Navonil Majumder, Soujanya Poria, Devamanyu Hazarika, Rada Mihalcea, Alexander Gelbukh, and Erik Cambria. 2019. Dialoguernn: An attentive rnn for emotion detection in conversations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6818–6825.
- Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.
- Soujanya Poria, Erik Cambria, Devamanyu Hazarika, Navonil Majumder, Amir Zadeh, and Louis-Philippe Morency. 2017. Context-dependent sentiment analysis in user-generated videos. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 873–883.
- Rohit Saxena, Savita Bhat, and Niranjan Pedanekar. 2018. Emotionx-area66: Predicting emotions in dialogues using hierarchical attention network with sequence labeling. In *Proceedings of the Sixth International Workshop on Natural Language Processing for Social Media*, pages 50–55.
- Björn Schuller, Michel Valster, Florian Eyben, Roddy Cowie, and Maja Pantic. 2012. Avec 2012: the continuous audio/visual emotion challenge. In *Proceedings of the 14th ACM international conference on Multimodal interaction*, pages 449–456. ACM.
- Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. 2015. End-to-end memory networks. In *Advances in neural information processing systems*, pages 2440–2448.
- Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1422–1432.
- Johnny Torres. 2018. Emotionx-jtml: Detecting emotions with attention. In *Proceedings of the Sixth International Workshop on Natural Language Processing for Social Media*, pages 56–60.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 1480–1489.

Tom Young, Erik Cambria, Iti Chaturvedi, Hao Zhou, Subham Biswas, and Minlie Huang. 2018. Augmenting end-to-end dialogue systems with common-sense knowledge. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

## A Appendices

Dataset	Partition	# of utterances	# of dialogues
IEMOCAP	train	5810	120
	test	1623	31
AVEC	train	4368	63
	test	1430	32

Table 3: Data distribution of IEMOCAP and AVEC datasets.

**Training Details** We use 10% of the training set as the validation set for hyper-parameters tuning. All tokens are lowercased with removal of stop words, symbols and digits, and sentences are zero-padded to the length of the longest sentence in the dataset. We alter the weight that each training instance carries when computing the loss to mitigate the influence of data imbalance. The weights are specific factors depending on corresponding emotions.

Hyper-parameters	IEMOCAP	AVEC
Optimizer	Adam	Adam
Learning rate	0.001	0.001
Batch size	16	16
Bi-GRU layer	2	2
Reasoning module layer	2	2
Historical utterance	30	20
GRU hidden size	150	150
Attention head	4	2
Attention hidden size	256	256

Table 4: Hyper-parameters and settings used for the two datasets.

# SentiRec: Sentiment Diversity-aware Neural News Recommendation

Chuhan Wu<sup>†</sup> Fangzhao Wu<sup>‡</sup> Tao Qi<sup>†</sup> Yongfeng Huang<sup>†</sup>

<sup>†</sup>Department of Electronic Engineering & BNRIst, Tsinghua University, Beijing 100084, China

<sup>‡</sup>Microsoft Research Asia, Beijing 100080, China

{wuchuhan15, wufangzhao, taoqi.qt}@gmail.com

yfhuang@tsinghua.edu.cn

## Abstract

Personalized news recommendation is important for online news services. Many news recommendation methods recommend news based on their relevance to users' historical browsed news, and the recommended news usually have similar sentiment with browsed news. However, if browsed news is dominated by certain kinds of sentiment, the model may intensively recommend news with the same sentiment orientation, making it difficult for users to receive diverse opinions and news events. In this paper, we propose a sentiment diversity-aware neural news recommendation approach, which can recommend news with more diverse sentiment. In our approach, we propose a sentiment-aware news encoder, which is jointly trained with an auxiliary sentiment prediction task, to learn sentiment-aware news representations. We learn user representations from browsed news representations, and compute click scores based on user and candidate news representations. In addition, we propose a sentiment diversity regularization method to penalize the model by combining the overall sentiment orientation of browsed news as well as the click and sentiment scores of candidate news. Extensive experiments on real-world dataset show that our approach can effectively improve the sentiment diversity in news recommendation without performance sacrifice.

## 1 Introduction

Online news websites such as Google news<sup>1</sup> have gained huge popularity for consuming digital news (Das et al., 2007). However, it is difficult for users to find their interested news information due to the huge volume of news emerging every day (Okura et al., 2017). Thus, personalized news recommendation is important for news websites to

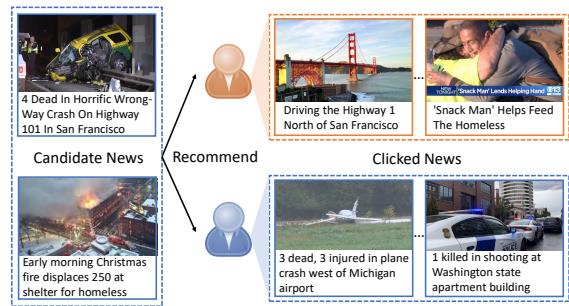


Figure 1: Several news browsed by two users of MSN News and the candidate news recommended to them.

target user interest and alleviate information overload (Wu et al., 2019a).

Many existing news recommendation methods rank candidate news based on their relevance to the interests of users inferred from their historical browsed news (Okura et al., 2017; Wu et al., 2019c). For example, Okura et al. (2017) proposed to learn news representations from news texts via autoencoders, and learn user representations from browsed news using a gated recurrent unit (GRU) network. They ranked candidate news based on the inner product of the user representation and candidate news representation. Wu et al. (2019c) proposed to learn news and user representations using multi-head self-attention networks. They ranked news based on the click scores computed by the dot product between news and user representations. The news articles recommended by these methods are usually similar to those previously browsed by a user in many aspects, such as content and sentiment. For example, in Fig. 1 the two candidate news articles are recommended to both users. The first user browses a news about the highway in San Francisco and a news about a person helping the homeless, which has inherent relatedness with the content of the candidate news. The second user browses several news about deadly accidents and

<sup>1</sup><https://news.google.com/>

crime, which has the same sentiment orientation as the candidate news. However, like the recommendations for the second user in Fig. 1, if a user mainly browses news articles that have a certain kind of sentiment (e.g., negative sentiment), many existing methods may intensively recommend news with the same sentiment orientation, which is not beneficial for this user to receive diverse opinions and news events that convey other sentiments.

In this paper, we propose a sentiment diversity-aware news recommendation approach named *SentiRec*, which can improve the sentiment diversity of news recommendation by considering the sentiment orientation of candidate and browsed news. In our approach, we propose a sentiment-aware news encoder, which is jointly trained with an auxiliary news sentiment prediction task, to incorporate sentiment information into news modeling and generate sentiment-aware news representations. We learn user representations from the representations of browsed news, and compute the click scores of candidate news based on their relevance to the user representations. In addition, to enhance the sentiment diversity of news recommendation, we propose a sentiment diversity regularization method to penalize our model during model training, which is based on the overall sentiment orientation of browsed news as well as the sentiment scores and click scores of candidate news. We conduct extensive experiments on a real-world benchmark dataset, and the results show that our approach can achieve better sentiment diversity and recommendation accuracy than many baseline methods.

The contributions of this paper are summarized as follows:

- To the best of our knowledge, this is the first work that explores to improve the sentiment diversity of news recommendation.
- We propose a sentiment-aware news encoder that incorporates an auxiliary news sentiment prediction task to encode sentiment-aware news representations.
- We propose a sentiment diversity regularization method to encourage the model to recommend news with diverse sentiment from the browsed news.
- Extensive experiments on real-world benchmark dataset verify that our approach can recommend news with diverse sentiment without performance loss.

## 2 Related Work

News recommendation is an important technique for online news websites to provide personalized news reading services (Zheng et al., 2018). A core problem in news recommendation is building accurate representations of news and users and further ranking candidate news according to news and user representations (Okura et al., 2017). In many news recommendation methods, news ranking is based on the representations of news and users built by manual feature engineering (Liu et al., 2010; Capelle et al., 2012; Son et al., 2013; Karkali et al., 2013; Garcin et al., 2013; Bansal et al., 2015; Ren et al., 2015; Chen et al., 2017; Zihayat et al., 2019). For example, Liu et al. (2010) proposed to use topic categories and interest features generated by a Bayesian model to build news and user representations. They ranked candidate news based on the product of a content-based score computed from news representations and a filter-based score computed by collaborative filtering. Son et al. (2013) proposed an Explicit Localized Semantic Analysis (ELSA) model for location-based news recommendation. They proposed to represent news and users by extracting topic and location features from Wikipedia pages, and ranked news based on the cosine distance between the representations of news and user. Lian et al. (2018) proposed to use various handcrafted features to represent news and users, such as title length, news categories, user profiles and features extracted from user behavior histories. They ranked candidate news based on the click scores computed by a neural factorization machine. However, these methods rely on manual feature engineering to build news and user representations, which usually necessitate massive expertise. In addition, handcrafted features may not be optimal in representing news content and user interest.

In recent years, several news recommendation methods based on deep learning techniques are proposed (Okura et al., 2017; Khattar et al., 2018; Wang et al., 2018; Wu et al., 2019a; An et al., 2019; Wu et al., 2019b,c; Ge et al., 2020). For example, Okura et al. (2017) proposed to learn first news representations from news bodies using autoencoders, and then learn representations of users from their clicked news with a GRU network. Candidate news are ranked based on the click scores computed by the dot products between news and user representations. Wang et al. (2018) proposed to learn news representations from news titles and

their entities via a knowledge-aware CNN network, and learn user representations from clicked news with a candidate-aware attention network. They ranked candidate news based on the click scores computed from the concatenation of news and user representations via a feed-forward neural network. Wu et al. (2019c) proposed to learn news and user representations with a combination of multi-head self-attention and additive attention networks. They also used dot product to compute click scores for news ranking. These methods tend to recommend news articles which are similar with the news users previously browsed (Lin et al., 2014). Thus, these methods may recommend news with similar sentiment orientation with those previously browsed by users, which is not beneficial for users to receive diverse news information. Different from these methods, our approach can effectively recommend news with diverse sentiment to users by incorporating sentiment information into news modeling via a sentiment-aware news encoder and regularizing the model based on the sentiment orientation of browsed and candidate news.

### 3 Our Approach

In this section, we first present the formal definitions of the problem explored in this paper, then introduce the details of our sentiment diversity-aware news recommendation (*SentiRec*) approach.

#### 3.1 Problem Definition

The problem studied in this paper is defined as follows. Given a user  $u$  with her news browsing history  $H = [D_1, D_2, \dots, D_N]$  and a set of candidate news<sup>2</sup>  $C = [D_1^c, D_2^c, \dots, D_P^c]$  ( $N$  and  $P$  respectively denote the number of browsed news and candidate news), the goal of the news recommendation model is to predict the personalized click scores  $[\hat{y}_1, \hat{y}_2, \dots, \hat{y}_P]$  of these candidate news, which are further used for ranking and display. We denote the sentiment labels of the browsed news and candidate news as  $[s_1, s_2, \dots, s_N]$  and  $[s_1^c, s_2^c, \dots, s_P^c]$ , respectively. In this paper we assume the sentiment labels are real values from -1 to 1, which indicate the sentiment polarity of news articles. We denote the overall sentiment orientation of browsed news as  $s$ . The sentiment diversity is defined as the differences between the sentiment orientation of recommended news and the overall sentiment

<sup>2</sup>The candidate news set is usually recalled from the entire news pool.

of browsed news.<sup>3</sup> The sentiment diversity of the news ranking results  $C'$  for the user  $u$  is measured by a function  $d = f(C', s)$ . The recommendation diversity is better if more top ranked news in  $C'$  have the different sentiment orientation with  $s$ .

#### 3.2 News Recommendation Framework

In this section, we introduce the general news recommendation framework of our *SentiRec* approach, as shown in Fig. 2. There are three core components in this framework for news recommendation, i.e., sentiment-aware (SA) news encoder, user encoder, and click predictor. The sentiment-aware news encoder aims to learn representations of news articles from their texts, where their sentiments are taken into consideration. We apply the sentiment-aware news encoder to the browsed news  $[D_1, D_2, \dots, D_N]$  and the candidate news  $D^c$  to encode their sentiment-aware representations, which are respectively denoted as  $[\mathbf{r}_1, \mathbf{r}_2, \dots, \mathbf{r}_N]$  and  $\mathbf{r}^c$ . The user encoder aims to learn representations of users from the sentiment-aware representations of their browsed news. Motivated by (Vaswani et al., 2017), we use Transformer to capture the relatedness between browsed news and learn a unified representation  $\mathbf{u}$  for each user. The click predictor aims to compute the personalized click scores of candidate news by measuring the relevance between user and candidate news representations. Following many previous works (Okura et al., 2017; Wu et al., 2019b), we use dot product to implement the click predictor, and the click score  $\hat{y}$  is predicted by  $\hat{y} = \mathbf{u}^\top \mathbf{r}^c$ .

#### 3.3 Sentiment-Aware News Encoder

In this section, we introduce the details of the sentiment-aware news encoders in our *SentiRec* approach. Its architecture is shown in Fig. 3. Motivated by the news encoder in (Wu et al., 2019c), we first use a word embedding layer to convert the sequence of words in a news title into a sequence of semantic vectors, and then use a Transformer (Vaswani et al., 2017) to capture the contexts of words and build a unified  $\mathbf{r}$  representation of news texts. However, the news representations directly learned by the Transformer are usually not sentiment-bearing. In fact, the sentiment information of news is very important for understanding

<sup>3</sup>We do not strictly require the recommendation results in an impression to be diverse in sentiment. We expect the sentiment of recommended news in a long term (e.g., multiple impressions in months) is diverse.

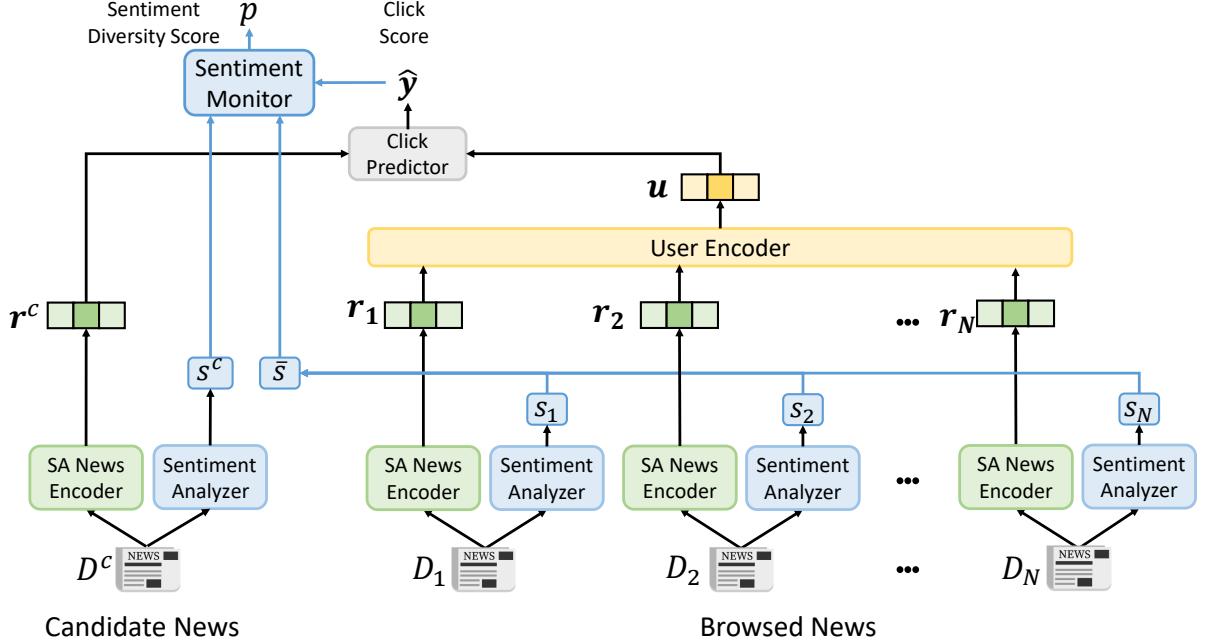


Figure 2: The framework of our *SentiRec* approach.

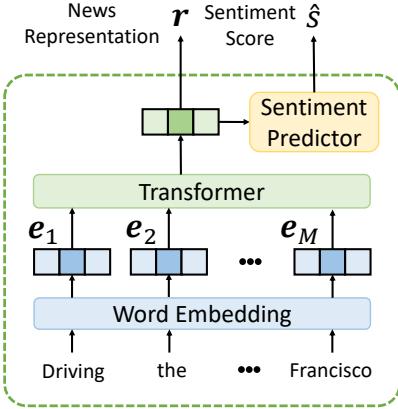


Figure 3: The architecture of the sentiment-aware news encoder.

the content of news. For example, in Fig. 1, although the news “Early morning...” and “Snack Man...” are both related to the homeless, they have opposite sentiment polarity, and modeling the sentiment of them can help understand their content better. In addition, the sentiment of news can also provide useful clues for user modeling and news ranking. For example, if a user frequently clicks negative news as the second user in Fig. 1, it may be more appropriate to recommend several positive news to this user rather than continuously recommending similar negative news. Thus, modeling news sentiment has the potential to enhance news recommendation. However, since the sentiment scores of news are numerical variables, simply re-

garding them as model input may be not optimal. Thus, we propose an auxiliary sentiment prediction task, and we jointly train the news encoder with this task to encourage it to learn sentiment-aware news representations. The real-valued sentiment score  $\hat{s}$  is predicted as follows:

$$\hat{s} = \mathbf{V}_s \times \mathbf{r} + \mathbf{v}_s, \quad (1)$$

where  $\mathbf{V}_s$  and  $\mathbf{v}_s$  are parameters. The loss function of sentiment prediction we use is the mean absolute error (MAE), which is formulated as follows:

$$\mathcal{L}_{senti} = \frac{1}{S} \sum_{i=1}^S |\hat{s}_i - s_i|, \quad (2)$$

where  $\hat{s}_i$  and  $s_i$  respectively stand for the predicted sentiment score and sentiment label of the  $i$ -th news, and  $S$  denotes the number of news. The sentiment labels are obtained by the sentiment analyzer modules in Fig. 2, which can be implemented by many sentiment analysis methods.

### 3.4 Sentiment Diversity Regularization

To further improve the sentiment diversity of news recommendation, we propose a sentiment diversity regularization method to penalize the recommendation model according to the overall sentiment score of browsed news, the sentiment score of candidate news, and its predicted click score. As shown in Fig. 2, we first use the sentiment analyzer to obtain the sentiment scores of the candidate

news (denoted as  $s^c$ ) and browsed news (denoted as  $[s_1, s_2, \dots, s_N]$ ). We then compute an overall sentiment score<sup>4</sup> of browsed news to indicate the historical sentiment preference of a user as follows:

$$\bar{s} = \frac{1}{N} \sum_{i=1}^N s_i. \quad (3)$$

A positive  $\bar{s}$  indicates that the user has read news with more positive sentiment and a negative  $\bar{s}$  indicates the negative sentiment is dominant in the browsed news. If the news recommender intensively recommends news with the same sentiment polarity with the overall sentiment  $s$  of a user’s browsed news, it is difficult for this user to receive diverse news information. Thus, it is important to recommend news with diverse sentiment to users. To solve this problem, we propose a sentiment diversity regularization method. We first propose to compute a sentiment diversity score  $p$  with a sentiment monitor, which is formulated as follows:

$$p = \max(0, \bar{s}s^c\hat{y}), \quad (4)$$

where a larger score of  $p$  indicates less sentiment diversity. In this formula, for a candidate news that shares the same sentiment polarity with  $s$ , the score  $p$  is larger if the model assigns it a higher click score or its sentiment and the overall browsed news sentiment are more intense, which indicate that the recommendation is less diverse in sentiment. Then, we propose a sentiment diversity loss function to regularize our model as follows:

$$\mathcal{L}_{div} = \frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} p_i, \quad (5)$$

where  $\mathcal{S}$  is the data set for model training, and  $p_i$  denotes the sentiment diversity score of the  $i$ -th sample in  $\mathcal{S}$ .

### 3.5 Model Training

In this section, we introduce how to train the models in our *SentiRec* approach. Following (Huang et al., 2013; Wu et al., 2019c), we use negative sampling techniques to construct labeled data for the news recommendation task from the user impression logs. More specifically, for each news clicked by a user, we randomly sample  $K$  news displayed in the same impression which are not clicked by

<sup>4</sup>We do not incorporate the numbers of positive and negative news because they cannot take the sentiment intensity into consideration.

this user. We denote the click scores of the  $i$ -th clicked news as  $\hat{y}_i^+$  and the associated  $K$  non-click news as  $[\hat{y}_{i,1}^-, \hat{y}_{i,2}^-, \dots, \hat{y}_{i,K}^-]$ . we use the click predictor to jointly predict these scores, and normalize these scores via the softmax function to compute the click probability scores. The news recommendation loss we used is the negative log-likelihood of the clicked news samples, which is computed as:

$$\mathcal{L}_{rec} = \sum_{i \in \mathcal{S}} \log\left(\frac{\exp(\hat{y}_i^+)}{\exp(\hat{y}_i^+) + \sum_{j=1}^K \exp(\hat{y}_{i,j}^-)}\right), \quad (6)$$

where  $\mathcal{S}$  is the data set for model training. We jointly train the news recommendation model with the auxiliary sentiment prediction task and meanwhile regularize it using the sentiment diversity loss. The final unified loss function of our approach is a weighted summation of the three loss functions, which is formulated as follows:

$$\mathcal{L} = \mathcal{L}_{rec} + \lambda \mathcal{L}_{senti} + \mu \mathcal{L}_{div}, \quad (7)$$

where  $\lambda$  and  $\mu$  are coefficients to control the relative importance of the sentiment prediction loss and sentiment diversity regularization loss.

## 4 Experiments

### 4.1 Datasets and Experimental Settings

Our experiments were conducted on a real-world news recommendation dataset provided by (Wu et al., 2019b), which is constructed from MSN News<sup>5</sup> logs from Oct. 31, 2018 to Jan. 29, 2019. We use the logs in the last week as the test set and the rest are used for training and validation, where the split ratio is 9:1.<sup>6</sup> To obtain the sentiment labels of the news in this dataset, we use the VADER algorithm (Hutto and Gilbert, 2014) as the sentiment analyzer in our approach.<sup>7</sup> It is a famous sentiment analysis method based on a set of sentiment lexicons such as LIWC (Pennebaker et al., 2001), ANEW (Nielsen, 2011) and GI (Stone et al., 1966). We use VADER to compute an overall sentiment orientation score of each news as the gold label, and these scores are ranged in [-1, 1]. The detailed statistics of the news recommendation dataset are shown in Table 1. We also plot the distribution of news sentiment scores and the overall sentiment orientation of users’ browsed news

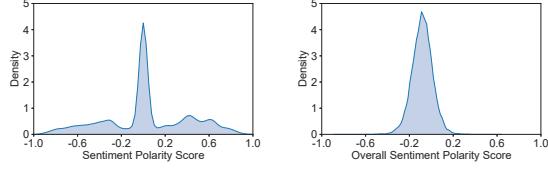
<sup>5</sup><https://www.msn.com/en-us/news>

<sup>6</sup>The numbers of constructed samples for training and validation are 277,811 and 30,868, respectively. The number of samples for test is 1,707,588.

<sup>7</sup>We choose this algorithm because it can compute the real-valued sentiment scores rather than polarity only.

# users	10,000	avg. # words per title	11.29
# news	42,255	# click samples	489,644
# impressions	445,230	# non-click samples	6,651,940
# samples	7,141,584	avg. sentiment score	0.0314

Table 1: Statistics of the dataset.



(a) News sentiment polarity (b) Overall sentiment orientation of users’ browsed news.

Figure 4: Distributions of the news sentiment polarity scores and the overall sentiment orientation of users’ browsed news in our dataset.

in Figs. 4(a) and 4(b), respectively. We find that although positive and negative news are almost balanced, the overall sentiment orientation of users’ browsed news is negative. In addition, we show the average click-through rate (CTR) of news with different ranges of sentiment scores in Fig. 5. As the saying goes, “*evil news rides fast, while good news baits later*”. We find it is interesting that more negative news have higher CTRs, which indicates that negative news has stronger ability in attracting news clicks. Thus, it is important to recommend news with diverse sentiment to avoid overwhelming users with too much negative news information.

Following Wu et al. (2019b), in our experiments the word embeddings were initialized by the 300-dimensional Glove embeddings (Pennington et al., 2014). The negative sampling ratio  $K$  was set to 4. Adam (Kingma and Ba, 2014) was chosen as the optimizer and the size of a minibatch was 30. In addition, the loss weights  $\lambda$  and  $\mu$  were respectively set to 0.4 and 10. These hyperparameters were tuned on the validation set. To evaluate the performance of news recommendation, we use metrics including AUC, MRR, nDCG@5 and nDCG@10<sup>8</sup>.

Since there is no off-the-shelf metric to evaluate the sentiment diversity of news recommendation, motivated by the MRR and hit ratio metrics, we propose three metrics named  $Senti_{MRR}$ ,  $Senti@5$  and  $Senti@10$  to quantitatively measure sentiment

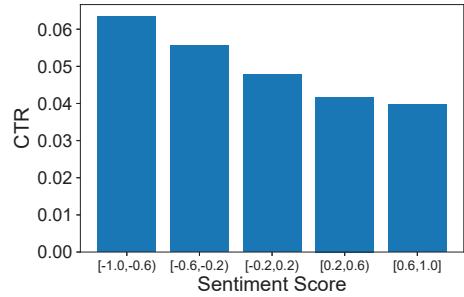


Figure 5: Click-through rates of news with different sentiment polarity scores.

diversity. They are computed as follows:

$$\begin{aligned} Senti_{MRR} &= \max(0, \bar{s} \sum_{i=1}^C \frac{s_i^c}{i}), \\ Senti@5 &= \max(0, \bar{s} \sum_{i=1}^5 s_i^c), \\ Senti@10 &= \max(0, \bar{s} \sum_{i=1}^{10} s_i^c), \end{aligned} \quad (8)$$

where  $C$  is the number of candidate news in an impression,  $s_i^c$  denotes the sentiment score of the candidate news with the  $i$ -th highest click score. In these metrics, higher scores indicate that the recommendation results are less diverse from the browsed news in their sentiment.<sup>9</sup> We repeated each experiment 10 times and reported the average results over all impressions in terms of the recommendation performance and sentiment diversity.

## 4.2 Performance Evaluation

We evaluate the recommendation performance and sentiment diversity of our approach by comparing it with several baseline methods, including: (1) *LibFM* (Rendle, 2012), a feature-based recommendation method based on factorization machine. TF-IDF features are used to represent the textual content of news. (2) *EBNR* (Okura et al., 2017), an embedding-based neural news recommendation method. It uses denoising autoencoders to learn news representations and a GRU network to encode user representations. (3) *DKN* (Wang et al., 2018), a knowledge-aware news recommendation method, which learns news representations via knowledge-aware CNN networks and learns user representations with a candidate-aware attention network.

<sup>8</sup>The relevance grade is binary, i.e., 0 for non-clicked news and 1 for clicked news.

<sup>9</sup>The scores are positive if the top ranked news have the same sentiment orientation with the overall sentiment, and are higher if these sentiments are more intensive.

Methods	AUC	MRR	nDCG@5	nDCG@10
LibFM	0.5661	0.2414	0.2689	0.3552
EBNR	0.6102	0.2811	0.3035	0.3952
DKN	0.6032	0.2744	0.2967	0.3873
Conv3D	0.6051	0.2765	0.2987	0.3904
DAN	0.6154	0.2860	0.3093	0.3996
NPA	0.6240	0.2952	0.3185	0.4094
NAML	0.6205	0.2902	0.3144	0.4060
NRMS	0.6275	0.2985	0.3217	0.4139
SentiRec	0.6294	0.3013	0.3237	0.4165
SentiRec-same	0.6299	0.3017	0.3240	0.4171

Table 2: Results of recommendation performance. Higher scores indicate better performance.

(4) *Conv3D* (Khattar et al., 2018), a neural news recommendation method which learns news representations using 2-D CNN models and learns user representations using a 3-D CNN model. (5) *DAN* (Zhu et al., 2019), a neural news recommendation method which learns news representations from title and entities with two independent CNN models and learns user representations using attentive LSTM network. (6) *NPA* (Wu et al., 2019b), a neural news recommendation method which learns news and user representations via personalized attention mechanism. (7) *NAML* (Wu et al., 2019a), a neural news recommendation method which learns news representations with CNN models and learns user representations using attention networks. (8) *NRMS* (Wu et al., 2019c), a neural news recommendation method which learns news and representations using multi-head self-attention and additive attention networks. For fair comparison, in all methods we used news titles to learn news representations. In addition, we compare the sentiment diversity of random news ranking, which aims to show the benchmark sentiment diversity without news and user modeling. Besides, we also compare a variant of our method (denoted as *SentiRec-same*) which only recommends the news with the same sentiment polarity with the browsed news (filter the candidate news with different sentiment polarity), which aims to show the scores of an extreme case with minimal sentiment diversity.

The results of recommendation performance and sentiment diversity are summarized in Tables 2 and 3. From these results, we find that neural news recommendation approaches achieve better recommendation performance than *LibFM*. This is probably because neural networks can learn more informative news and user representations than traditional matrix factorization methods. However, compared with random ranking, we find that the diver-

Methods	<i>Senti<sub>MRR</sub></i>	<i>Senti@5</i>	<i>Senti@10</i>
Random	0.0262	0.0442	0.0687
LibFM	0.0843	0.1192	0.2579
EBNR	0.0989	0.1476	0.2868
DKN	0.0954	0.1389	0.2810
Conv3D	0.0973	0.1431	0.2830
DAN	0.1005	0.1520	0.2897
NPA	0.1044	0.1583	0.3015
NAML	0.1030	0.1569	0.2967
NRMS	0.1066	0.1592	0.3034
SentiRec	<b>0.0046</b>	<b>0.0083</b>	<b>0.0115</b>
SentiRec-same	0.3271	0.4963	0.9373

Table 3: Results of sentiment diversity. Lower scores indicate better sentiment diversity.

sity scores of all baseline methods are much larger, especially those based on neural networks. This is probably because the compared baseline methods mainly recommend news based on the relevance between candidate news and browsed news, and will tend to recommend news with similar sentiment orientation with browsed news, which is harmful for users to receive diverse news information. Different from baseline methods, our *SentiRec* approach can achieve much better sentiment diversity even than random ranking. These results show that our approach can actively recommend news with diverse sentiment from browsed news. In addition, our approach can also achieve better recommendation performance than baseline methods. These results validate that our approach can achieve the goal of improving sentiment diversity in news recommendation without hurting the recommendation performance. Besides, by comparing *SentiRec* and its variant *SentiRec-same*, although the recommendation performance of *SentiRec-same* is slightly better, the sentiment of its recommendation results are minimally diverse from browsed news, which may amplify the problem of filter bubble and hurt user experience.

### 4.3 Ablation Study

In this section, we conduct ablation studies to verify the influence of the auxiliary sentiment prediction task in the sentiment-aware news encoder and the sentiment diversity regularization method on the recommendation performance and sentiment diversity. The results are shown in Fig. 6. From Fig. 6, we find that the sentiment prediction task can improve both sentiment diversity and recommendation performance. This may be because this auxiliary task can encourage the news encoder to

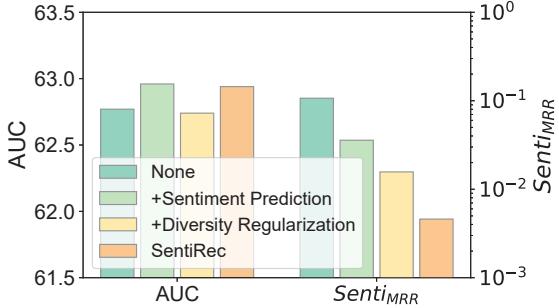


Figure 6: Influence of the sentiment prediction task and sentiment diversity regularization method.

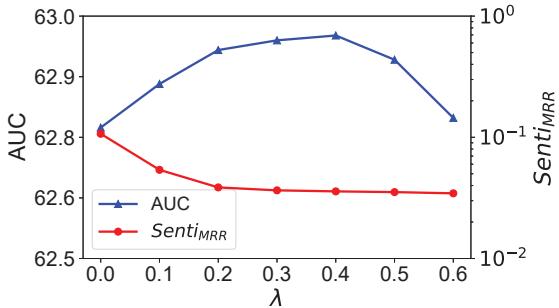


Figure 7: Influence of the hyperparameter  $\lambda$ .

exploit the sentiment information in news texts to encode sentiment-aware news representations, which is beneficial for predicting news clicks more accurately and further improving sentiment diversity by modeling users’ dynamic preferences on news sentiment. In addition, the sentiment diversity regularization can also effectively improve the sentiment diversity of news recommendation and meanwhile keep the recommendation performance. This is because this regularization method can enforce the model to recommend news with different sentiment orientations with the browsed news. Moreover, combining both techniques can further improve sentiment diversity, which verifies the effectiveness of our *SentiRec* method.

#### 4.4 Influence of Hyperparameters

In this section, we will explore the influence of two important hyperparameters on our approach, i.e., the loss coefficients  $\lambda$  and  $\mu$  in Eq. (7) on the performance and sentiment diversity of our approach. Since there are two hyperparameters, we first vary the value of  $\lambda$  to find the optimal one to learn sentiment-aware news representations in our approach. The results are illustrated in Fig. 7. According to Fig. 7, we find both sentiment diversity and recommendation performance of our approach

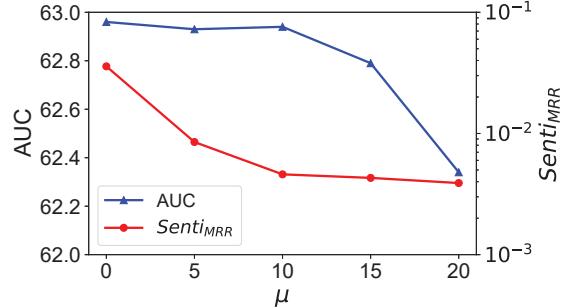


Figure 8: Influence of  $\mu$  under  $\lambda = 0.4$ .

improves when  $\lambda$  increases from 0. This is probably because when  $\lambda$  is too small, the useful sentiment information in news cannot be fully exploited. However, the performance of our approach starts to decline when  $\lambda$  is too large. This may be because when  $\lambda$  is too large, the auxiliary sentiment prediction task is over-emphasized and the news recommendation task is not fully respected. Thus, a moderate  $\lambda$  (e.g., 0.4) is more appropriate for our approach to make a tradeoff between recommendation performance and sentiment diversity.

Then, we vary the value of  $\mu$  under  $\lambda = 0.4$  to evaluate the recommendation performance and sentiment diversity of our approach.<sup>10</sup> The results are illustrated in Fig. 8. According to the results, we find the sentiment diversity can be consistently improved when  $\mu$  increases. This is probably because when  $\mu$  is larger, the model is regularized more intensively and may tend to recommend more news with diverse sentiment from browsed news. However, when  $\mu$  goes too large, the performance in terms of AUC declines significantly, which may hurt user experience. Thus, a moderate selection on  $\mu$  (e.g., 10) is appropriate to achieve the goal of recommending news with diverse sentiment and meanwhile keep good recommendation performance.

#### 4.5 Case Study

In this section, we present several case studies to better demonstrate the effectiveness of our approach in improving sentiment diversity of news recommendation. The clicked news of a randomly selected user as well as the top ranked candidate news recommended by a state-of-the-art method *NRMS* and our *SentiRec* approach are shown in Table 4. We can see that the historical browsed news of this user are mainly about negative topics such

<sup>10</sup>We find that the scale of the regularization loss is relatively small and the magnitude of  $\mu$  needs to be larger.

Browsed News	Top Ranked Candidate News	
	NRMS	SentiRec
Woman Arrested for Alleged California Wildfire Scam	Sheriff: California officer’s killer is in the US illegally	Eight 2018 Fashion Trends We’re Ready to Move On From
Guns are the second leading killer of kids, after cars	Professional golfer and his caddie arrested for poaching at a tiger reserve	Josh Duhamel Wants to Date Someone Young Enough to Have Kids
From international fashion model to suspect in racist attack on Kansas toddler	Trump threatens years-long shutdown for his wall as GOP support begins to fracture	58 Amazing After-Christmas Deals Happening Right Now

Table 4: The browsed news of a user and top ranked candidate news provided by different methods.

as crime, which usually convey negative sentiment. However, the *NRMS* method still intensively recommends news with negative sentiment such as “Sheriff: California officer’s killer...”. It indicates that *NRMS* tends to recommend news with similar sentiment to the browsed news, which is not suitable for users to acquire diverse news information. Different from *NRMS*, our approach can effectively recommend news with diverse sentiment from browsed news, and the recommended news also has some inherent relatedness with browsed news in their content (e.g., both the first candidate news and the third browsed news mention “fashion”). It shows that our approach can improve the sentiment diversity of news recommendation and meanwhile keep recommendation accuracy.

## 5 Conclusion and Future Work

In this paper, we propose a sentiment diversity-aware neural news recommendation approach which can effectively recommend news with diverse sentiment from browsed news. We propose a sentiment-aware news encoder to learn sentiment-aware news representations by jointly training it with an auxiliary sentiment prediction task. We learn user representations from representations of browsed news, and compute click scores based on user and candidate news representations. In addition, we propose a sentiment diversity regularization method to regularize the model according to the overall sentiment orientation of browsed news as well as the click scores and sentiment scores of candidate news. Extensive experiments on real-world benchmark dataset validate that our approach can effectively enhance the sentiment diversity of news recommendation without hurting the recommendation performance.

In our future work, we plan to analyze the sentiment on the entities in news and explore to improve the entity-level sentiment diversity of news recommendation. In addition, we plan to extend sentiment polarities to more kinds of emotions, such as

angry, happiness, sad and surprise, to enhance the emotion diversity of news recommendation.

## Acknowledgments

This work was supported by the National Key Research and Development Program of China under Grant number 2018YFB2101501, and the National Natural Science Foundation of China under Grant numbers U1936208 and U1936216.

## References

- Mingxiao An, Fangzhao Wu, Chuhan Wu, Kun Zhang, Zheng Liu, and Xing Xie. 2019. Neural news recommendation with long-and short-term user representations. In *ACL*, pages 336–345.
- Trapit Bansal, Mrinal Das, and Chiranjib Bhattacharyya. 2015. Content driven user profiling for comment-worthy recommendations of news and blog articles. In *RecSys.*, pages 195–202. ACM.
- Michel Capelle, Flavius Frasincar, Marnix Moerland, and Frederik Hogenboom. 2012. Semantics-based news recommendation. In *WIMS*, page 27. ACM.
- Cheng Chen, Xiangwu Meng, Zhenghua Xu, and Thomas Lukasiewicz. 2017. Location-aware personalized news recommendation with deep semantic analysis. *IEEE Access*, 5:1624–1638.
- Abhinandan S Das, Mayur Datar, Ashutosh Garg, and Shyam Rajaram. 2007. Google news personalization: scalable online collaborative filtering. In *WWW*, pages 271–280. ACM.
- Florent Garcia, Christos Dimitrakakis, and Boi Faltings. 2013. Personalized news recommendation with context trees. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 105–112. ACM.
- Suyu Ge, Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang. 2020. Graph enhanced representation learning for news recommendation. In *WWW*, pages 2863–2869.
- Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using

- clickthrough data. In *CIKM*, pages 2333–2338. ACM.
- Clayton J Hutto and Eric Gilbert. 2014. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *ICWSM*.
- Margarita Karkali, Dimitris Pontikis, and Michalis Vazirgiannis. 2013. Match the news: A firefox extension for real-time news recommendation. In *SIGIR*, pages 1117–1118. ACM.
- Dhruv Khattar, Vaibhav Kumar, Vasudeva Varma, and Manish Gupta. 2018. Weave& rec: A word embedding based 3-d convolutional network for news recommendation. In *CIKM*, pages 1855–1858. ACM.
- Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Jianxun Lian, Fuzheng Zhang, Xing Xie, and Guangzhong Sun. 2018. Towards better representation learning for personalized news recommendation: a multi-channel deep fusion approach. In *IJCAI*, pages 3805–3811.
- Chen Lin, Runquan Xie, Xinjun Guan, Lei Li, and Tao Li. 2014. Personalized news recommendation via implicit social experts. *Information Sciences*, pages 1–18.
- Jiahui Liu, Peter Dolan, and Elin Rønby Pedersen. 2010. Personalized news recommendation based on click behavior. In *IUI*, pages 31–40. ACM.
- Finn Årup Nielsen. 2011. A new anew: Evaluation of a word list for sentiment analysis in microblogs. In *Workshop on Making Sense of Microposts: Big things come in small packages*, pages 93–98.
- Shumpei Okura, Yukihiro Tagami, Shingo Ono, and Akira Tajima. 2017. Embedding-based news recommendation for millions of users. In *KDD*, pages 1933–1942. ACM.
- James W Pennebaker, Martha E Francis, and Roger J Booth. 2001. Linguistic inquiry and word count: Liwc 2001. *Mahway: Lawrence Erlbaum Associates*, 71(2001):2001.
- Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543.
- Rui Ren, Lingling Zhang, Limeng Cui, Bo Deng, and Yong Shi. 2015. Personalized financial news recommendation algorithm based on ontology. *Procedia Computer Science*, 55:843–851.
- Steffen Rendle. 2012. Factorization machines with libfm. *TIST*, 3(3):57.
- Jeong-Woo Son, A Kim, Seong-Bae Park, et al. 2013. A location-based news article recommendation with explicit localized semantic analysis. In *SIGIR*, pages 293–302. ACM.
- Philip J Stone, Dexter C Dunphy, and Marshall S Smith. 1966. The general inquirer: A computer approach to content analysis.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*, pages 5998–6008.
- Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. Dkn: Deep knowledge-aware network for news recommendation. In *WWW*, pages 1835–1844.
- Chuhan Wu, Fangzhao Wu, Mingxiao An, Jianqiang Huang, Yongfeng Huang, and Xing Xie. 2019a. Neural news recommendation with attentive multi-view learning. In *IJCAI*, pages 3863–3869. AAAI Press.
- Chuhan Wu, Fangzhao Wu, Mingxiao An, Jianqiang Huang, Yongfeng Huang, and Xing Xie. 2019b. Npa: Neural news recommendation with personalized attention. In *KDD*, pages 2576–2584. ACM.
- Chuhan Wu, Fangzhao Wu, Suyu Ge, Tao Qi, Yongfeng Huang, and Xing Xie. 2019c. Neural news recommendation with multi-head self-attention. In *EMNLP-IJCNLP*, pages 6390–6395.
- Guanjie Zheng, Fuzheng Zhang, Zihan Zheng, Yang Xiang, Nicholas Jing Yuan, Xing Xie, and Zhen-hui Li. 2018. Drn: A deep reinforcement learning framework for news recommendation. In *WWW*, pages 167–176.
- Qiannan Zhu, Xiaofei Zhou, Zeliang Song, Jianlong Tan, and Li Guo. 2019. Dan: Deep attention neural network for news recommendation. In *AAAI*, volume 33, pages 5973–5980.
- Morteza Zihayat, Anteneh Ayanso, Xing Zhao, Heidar Davoudi, and Aijun An. 2019. A utility-based news recommendation system. *Decision Support Systems*, 117:14–27.

# BCTH: A Novel Text Hashing Approach via Bayesian Clustering

Wenjie Ying<sup>†,\*</sup>, Yuquan Le<sup>‡,\*</sup>, Hantao Xiong<sup>‡</sup>

<sup>†</sup> Baidu Inc., Beijing, China

<sup>‡</sup> Changsha Lvzhidao Information Technology Co., Ltd., Changsha, China

yingwenjie@baidu.com, leyuquan@yeah.net, xionghantao94@163.com

## Abstract

Similarity search is to find the most similar items for a certain target item. The ability of similarity search at large scale plays a significant role in many information retrieval applications and has received much attention. Text hashing is a promising strategy, which utilizes binary encoding to represent documents, and is able to obtain attractive performance. This paper makes the first attempt to utilize Bayesian Clustering for Text Hashing, dubbed as BCTH. Specifically, BCTH can map documents to binary codes by utilizing multiple Bayesian Clusterings in parallel, where each Bayesian Clustering is responsible for one bit. Our approach employs the bit-balanced constraint to maximize the amount of information in each bit. Meanwhile, the bit-uncorrelated constraint is adopted to keep independence among all bits. The time complexity of BCTH is linear, where the hash codes and hash functions are jointly learned. Based on four widely-used datasets, the experimental results demonstrate that BCTH is competitive compared with currently competitive baselines from the perspective of both precision and training speed.

## 1 Introduction

The task of *similarity search*, also called *nearest neighbor search*, aims to find the most similar objects for a given query item (Gionis et al., 1999; Andoni and Indyk, 2006). It plays a significant role in many information retrieval applications, such as document clustering, content-based retrieval, collaborative filtering (Wang et al., 2016), etc. With the development of many intelligent terminals, massive textual data has been produced over the past several decades. Huge challenges exist in applying text similarity algorithms (Conneau et al., 2017; Le et al., 2018) to large-scale corpora, since these

\*means the corresponding author.

methods require complicated numerical computation.

Text hashing (Severyn and Moschitti, 2015) is a promising strategy and has obtained much attention. It maps semantically similar documents to hash codes with similar semantics through designing binary codes in a low-dimensional space. A hashing representation of each document usually needs only a few bits to be stored. The calculation of the similarity between two hash codes can be executed by a bit-wise XOR operation. Therefore, text hashing is an effective strategy to accelerate similarity queries and reduce data storage.

Most of the traditional text hashing methods consist of two stages (Zhang et al., 2010; Lin et al., 2014b; Severyn and Moschitti, 2015). The first step is to learn hash code, preserving similarity among neighbors. Then the hash function is trained through the self-taught method, with the text features and hash codes as the input (Wang et al., 2013b). However, for  $m$  documents,  $O(m^2)$  training time complexity is needed to generate the pairwise similarity matrix used to preserve the similarity information. On the other hand, due to the success of deep learning, researchers have attempted to study text hashing through deep neural networks (Xu et al., 2015). Some of the most representative works include VDSH (Chaidaroon and Fang, 2017) and NASH (Kalchbrenner et al., 2014). The NASH model studies text hashing through an end-to-end Neural Architecture, which treats the hash codes as the latent factor. The VDSH model introduces a latent factor for documents to capture the semantic information. Even though these methods have achieved attractive performance, the training time is unsatisfactory, making them unscalable to large-scale datasets.

Motivated by the above observations, this paper attempts to utilize Bayesian Clustering for Text Hashing, dubbed as BCTH. Specifically, BCTH

can map documents to binary codes by using multiple Bayesian Clusterings in parallel, where each Bayesian Clustering is responsible for one bit. Our approach employs the bit-balanced constraint to maximize the amount of information in each bit. Meanwhile, the bit-uncorrelated constraint is adopted to keep independence among all bits. Experimental results prove that our approach is competitive in the perspective of both precision and training speed.

Our contributions are summarized as follows:

- We propose a novel Text Hashing based on the Bayesian Clustering framework, dubbed as BCTH, for learning effective hash codes from documents. To the best of our knowledge, this is the first work that utilizes Bayesian Clustering in text hashing.
- The time complexity of our method is linear, where the hash codes and hash function are jointly learned. What's more, we visualize the hash codes and prove that BCTH can obtain effective semantics from the original documents.
- We conduct extensive experiments on four public text datasets. Based on four widely-used datasets, the experimental results demonstrate that BCTH is competitive compared with currently competitive baselines from the perspective of both precision and training speed.

## 2 Model

The approach of our proposed BCTH is introduced in this section. As is shown in Fig. 1, BCTH is a general learning idea, which utilizes Bayesian Clustering that is based on the latent factor framework in Text Hashing. BCTH can map documents to binary codes by using multiple Bayesian Clusterings in parallel, where each Bayesian Clustering is responsible for one bit. During this process, the bit-balanced constraint is to maximize the amount of information in each bit. Meanwhile, the bit-uncorrelated constraint is adopted to keep independence among all bits.

### 2.1 Preliminaries

Given a set of  $m$  documents  $\mathbf{X} = \{x^{(i)}\}_{i=1}^m$ , where  $x^{(i)}$  is the feature representation of the  $i$ -th document. The binary code for the  $i$ -th document is

expressed as  $b^{(i)} = \{b_k^{(i)}, b_k^{(i)} \in \{-1, 1\}\}_{k=1}^r$ , and  $r$  is the length of the hash codes. Unlike the existing approaches (Liu et al., 2011; Zhang et al., 2010; Xu et al., 2015) that aim to preserve the pair-wise similarity among all the documents, we use Naive Bayes to extract the semantic information of the  $i$ -th document as:

$$P(b_k^{(i)} = c_k | x^{(i)}) = \frac{P(x^{(i)} | b_k^{(i)} = c_k) P(b_k^{(i)} = c_k)}{P(x^{(i)})} \quad (1)$$

The Naive Bayes method assumes the conditional independence for the conditional probability distribution, and therefore, we obtain the following equation:

$$P(x^{(i)} | b_k^{(i)} = c_k) = \prod_{j=1}^n P(w_j = l_j^{(i)} | b_k^{(i)} = c_k) \quad (2)$$

where  $c_k$  represents the  $k$ -th bit's value of the hash codes of the  $i$ -th document,  $c_k \in \{-1, 1\}$ , and  $n$  is the size of the vocabulary. The  $l_j^{(i)}$  denotes whether the  $j$ -th word of the vocabulary appears in the  $i$ -th document, and  $l_j^{(i)} \in \{0, 1\}$ .

The previous formula adopts the cumulative multiplication of all words' probabilities to calculate the likelihood of a particular document. However, since many words will not appear in a specific document, to avoid redundant calculation, we consider using the cumulative multiplication of the probabilities of words that appear in that particular document to calculate the probability of that document.

$$P(x^{(i)} | b_k^{(i)} = c_k) = \prod_{j \in \Phi^{(i)}} P(w_j = 1 | b_k^{(i)} = c_k) \quad (3)$$

In the above equation,  $\Phi^{(i)}$  is a set of words that appear in the  $i$ -th document. Each hash code can be learned through an unsupervised iteration process. By utilizing multiple Bayesian Clusterings to calculate all hash codes of documents in parallel, we obtain the following objective function:

$$P(\mathbf{B} | \mathbf{X}) = \frac{P(\mathbf{X} | \mathbf{B}) P(\mathbf{B})}{P(\mathbf{X})} \quad (4)$$

where hash codes of documents are expressed as  $\mathbf{B} = \{b_k^{(i)}, k = 1, 2..r, i = 1, 2.., m\}$ .

In order to obtain high-quality hash codes, the bit-balanced and the bit-uncorrelated constraints are introduced. In addition, we transform the probability from the interval  $[0, 1]$  to the interval  $[-1, 1]$

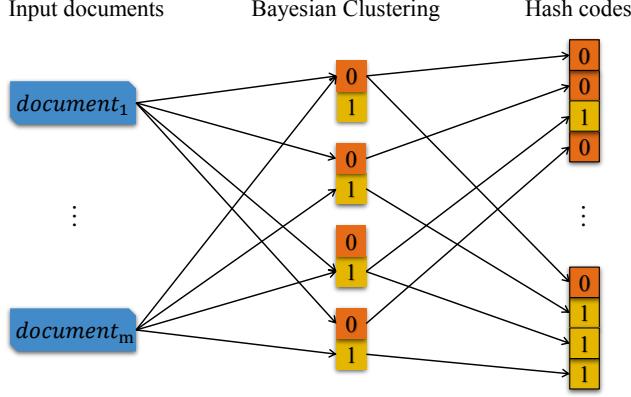


Figure 1: Illustration of how to learn the hash codes through multiple Bayesian Clusterings jointly from  $m$  documents. The size of hash codes in the illustration is  $r = 4$ .

by the function  $f(P) = 2P - 1$ . Therefore, we obtain the following loss function:

$$\begin{aligned} \min & \sum_{k=1}^r \sum_{i=1}^m \left\| b_k^{(i)} - p_k^{(i)} \right\|^2 \iff \|\mathbf{B} - \mathbf{P}\| \\ \text{s.t.} & \mathbf{B} \in \{-1, 1\}^{r \times m} \\ & \mathbf{B}\mathbf{1} = \mathbf{0}, \mathbf{B}\mathbf{B}^T = m\mathbf{I}_{r \times r} \end{aligned} \quad (5)$$

where  $p_k^{(i)} = f(P(b_k^{(i)} = c_k | x^{(i)}))$  and  $\mathbf{P} = \{p_k^{(i)}, k = 1, 2..r, i = 1, 2.., m\}$ . The  $\mathbf{1}$  denotes a vector with all of its elements equal to 1. The equality  $\mathbf{B}\mathbf{1} = \mathbf{0}$  denotes the bit-balanced constraint, which aims to maximize the amount of information in each bit. The equality  $\mathbf{B}\mathbf{B}^T = m\mathbf{I}_{r \times r}$  denotes the bit-uncorrelated constraint, aiming to keep the independence among all bits.

However, the Eq. (5) is difficult to solve directly. Following the prior work in discrete graph hashing (Liu et al., 2014), let us define the constraint space as  $\Omega = \{\mathbf{Y} \in \mathbf{R}^{r \times m} | \mathbf{Y}\mathbf{1} = \mathbf{0}, \mathbf{Y}\mathbf{Y}^T = m\mathbf{I}_{r \times r}\}$ . Then we formulate a more general framework which softens the two hard constraints in Eq. (5) as:

$$\begin{aligned} \min & \|\mathbf{B} - \mathbf{P}\|^2 + \lambda \|\mathbf{B} - \mathbf{Y}\|^2 \\ \text{s.t.} & \mathbf{B} \in \{-1, 1\}^{r \times m} \\ & \mathbf{Y}\mathbf{1} = \mathbf{0}, \mathbf{Y}\mathbf{Y}^T = m\mathbf{I}_{r \times r} \end{aligned} \quad (6)$$

where  $\lambda \geq 0$  is a hyper parameter and  $\mathbf{Y}$  is relaxation factor. If problem (5) is feasible, we can enforce  $\mathbf{B}\mathbf{1} = \mathbf{0}, \mathbf{B}\mathbf{B}^T = m\mathbf{I}_{r \times r}$  in Eq.(5) by setting an extremely large value to  $\lambda$ , thereby converting problem (6) into problem (5).

## 2.2 Learning

The learning process aims to find the desirable hash codes that can optimize the Eq. (6). Similar to (Liu

et al., 2014), we utilize a tractable alternating minimization algorithm, which is an unsupervised iteration process, including alternately solving three sub-problems.

**W-subproblem:** Let us initialize the hash codes  $\mathbf{B}$  randomly, and the parameter  $W = \{p(w_j = 1 | b_k = c_k), p(b_k = c_k)\}$ ,  $j \in \{1, 2, \dots, n\}$ ,  $k \in \{1, 2, \dots, r\}$  can be calculated by Naive Bayes. The document is represented by the one-hot method. The variable  $\mathbf{P}$  is calculated in the following way, specifically, through the conditional probability and prior probability. The formula is as follow:

$$p(b_k = c_k) = \frac{\sum_{i=1}^m I(b_k^{(i)} = c_k)}{m}, k \in \{1, 2, \dots, r\} \quad (7)$$

where the  $p(b_k = c_k)$  is the ratio of the number of documents with the  $k$ -th hash code equal to  $c_k$  to the total number of documents.  $I$  is the indicator function. If the input value is true, it returns 1, else returns 0. The calculation process of the conditional probability  $P(w_j = 1 | b_k = c_k)$ , which includes the strategy of Laplace smoothing, is as follow:

$$P(w_j = 1 | b_k = c_k) = \frac{\sum_{i=1}^m I(w_j^{(i)} = 1 \cap b_k^{(i)} = c_k) + 1}{\sum_{j=1}^n \sum_{i=1}^m I(w_j^{(i)} = 1 \cap b_k^{(i)} = c_k) + n} \quad (8)$$

where  $\sum_{i=1}^m I(w_j^{(i)} = 1 \cap b_k^{(i)} = c_k)$  is the number of documents whose  $k$ -th hash code value is  $c_k$ , which contains the word  $w_j$ . The "  $\cap$  " symbol means "and".

**Y-subproblem:** Given the value of  $\mathbf{B}$ , the continuous variable  $\mathbf{Y}$  can be calculated by Eq. (10),

the details are as follows:

$$\begin{aligned} \min_{\mathbf{Y}} \|\mathbf{B} - \mathbf{Y}\|^2 &\iff \min_{\mathbf{Y}} 2(m - \text{tr}(\mathbf{B}^T \mathbf{Y})) \\ \text{s.t. } \mathbf{Y}\mathbf{1} = \mathbf{0}, \mathbf{Y}\mathbf{Y}^T = m\mathbf{I}_{r \times r} \end{aligned} \quad (9)$$

Where  $\text{tr}$  is solving the trace of a matrix and  $\mathbf{I}_{r \times r}$  is an identity matrix.

Minimizing  $2(m - \text{tr}(\mathbf{B}^T \mathbf{Y}))$  is equivalent to maximizing the trace of the  $\mathbf{B}^T \mathbf{Y}$ , and it can be solved by performing singular value decomposition (SVD) operation on the matrix  $\bar{\mathbf{B}}$  where every element is calculated by:  $\bar{b}_k^{(i)} = b_k^{(i)} - \frac{1}{m} \sum_{i=1}^m b_k^{(i)}$ . The  $\mathbf{U}_b$  and  $\mathbf{V}_b$ , therein satisfying  $[\mathbf{V}_b \ \mathbf{1}]^T \hat{\mathbf{V}}_b = 0$ , are stacked by the left and right singular vectors respectively from the result of SVD. After performing Gram-Schmidt process on  $\mathbf{U}_b$  and  $\mathbf{V}_b$ , we obtain  $\mathbf{U}_b$  and  $\mathbf{V}_b$ . Finally, according to (Zhang et al., 2016), the  $\mathbf{Y}$  is updated by:

$$\mathbf{Y} = \sqrt{m} [\mathbf{U}_b \ \hat{\mathbf{U}}_b] [\mathbf{V}_b \ \hat{\mathbf{V}}_b]^T \quad (10)$$

**B-subproblem:** Given the value of  $\mathbf{P}$  and the continuous variable  $\mathbf{Y}$ , the value of  $\mathbf{B}$  can be calculated by minimizing Eq. (12), and the details are as follows:

$$\begin{aligned} \min_{\mathbf{B}} \|\mathbf{B} - \mathbf{P}\|^2 + \lambda \|\mathbf{B} - \mathbf{Y}\|^2 \\ \text{s.t. } \mathbf{B} \in \{-1, 1\}^{r \times m} \end{aligned} \quad (11)$$

Since Eq. (12) is a simple binary optimization process, we can update  $\mathbf{B}$  by updating each element of it in parallel according to:

$$\mathbf{b}_k^{(i)} = \underset{\mathbf{b}_k^{(i)} \in \{-1, 1\}}{\operatorname{argmin}} \left\| \mathbf{b}_k^{(i)} - \mathbf{p}_k^{(i)} \right\|^2 + \lambda \left\| \mathbf{b}_k^{(i)} - \mathbf{y}_k^{(i)} \right\|^2 \quad (12)$$

The whole algorithm implementation process is shown in algorithm 1.

### 2.3 Complexity Analysis

In this section, we analyze the space and time complexity of BCTH. The learning algorithm of BCTH is shown in Algorithm 1. For space complexity, Algorithm 1 requires  $O(mn + mr + nr)$  to store the training datasets, hash codes, and parameters. As  $r$  is usually less than 1024, we can easily store the above variables at large-scale in memory.

For time complexity, we first analyze each of the sub-problems. For  $\mathbf{W}$ -subproblem, it takes  $O(mnr)$  to calculate parameter  $\mathbf{W}$  and update

---

### Algorithm 1: Learning algorithm of BCTH

---

```

Input : Training data:  $\mathbf{X} \in \mathbf{R}^{m \times n}$ 
          code length:  $r$ 
          hyperparameter:  $\lambda$ ;
Output :  $W = \{p(w_j = 1 | b_k = c_k), p(b_k = c_k)\}, j \in \{1, 2, \dots, n\}, k \in \{1, 2, \dots, r\};$ 
1 Initialize:  $\mathbf{B}$  by randomization;
2 repeat
3 W-step:
4 Solve  $\mathbf{W}$  and  $\mathbf{P}$  in W-subproblem
5 Y-step:
6 Solve  $\mathbf{Y}$  in Y-subproblem
7 B-step:
8 Solve  $\mathbf{B}$  by B-subproblem
9 until convergence;
10 return  $\mathbf{W}, \mathbf{B};$ 

```

---

probability  $\mathbf{P}$ . For  $\mathbf{Y}$ -subproblem, it requires  $O(r^2n)$  to perform the SVD, Gram-Schmidt orthogonalization, and matrix multiplication. For  $\mathbf{B}$ -subproblem, it requires  $O(mn)$  to update each  $\mathbf{b}_k^{(i)}$  of  $\mathbf{B}$ . The time complexity of the whole Algorithm 1 is  $O(t(mnr + r^2n + mn))$ , where  $t$  is the number of iterations needed for convergence. In our experiments,  $t$  is set to 10 by default (See section 3.8). It can be seen that the time complexity of BCTH is linear.

## 3 Experiments

### 3.1 Datasets

Following prior works (Chaidaroon and Fang, 2017), we experiment on four public text datasets.

- **Reuters Corpus Volume I (RCV1):** The *RCV1* is a large collection of manually labeled 800,000 newswire stories provided by *Reuters*. The full-topics version is available at the LIBSVM website<sup>1</sup>.
- **Reuters21578 (Reuters):** This dataset is a widely-used text corpus for text classification. This collection contains 10,788 documents with 90 categories and 7,164 unique words.
- **TMC<sup>3</sup>:** This dataset has 22 labels, 21,519

<sup>1</sup><https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/multilabel.html>

<sup>2</sup><http://www.nltk.org/book/ch02.html>

<sup>3</sup><https://catalog.data.gov/dataset/siam-2007-text-mining-competition-dataset>

training set, 3,498 test set, and 3,498 documents for the validation set. This dataset is used as part of the SIAM text mining competition and contains the air traffic reports provided by NASA.

- **20Newsgroups<sup>4</sup>:** The 20 Newsgroups dataset is a collection of 18828 newsgroup documents. It is divided into different newsgroups, each corresponding to a specific topic.

### 3.2 Baselines and Evaluation Metrics

We compare BCTH with the following competitive unsupervised methods since BCTH also belongs to unsupervised methods.

- **LSH:** This approach applies (Datar et al., 2004) random projections as the hash function to transform the data points from its original space to the binary hash space. More hash bits are needed to guarantee the precision on account of the randomness of the hash function.
- **SH:** This baseline (Weiss et al., 2008) calculates the bits through thresholding a subset of eigenvectors of the Laplacian of the similarity graph.
- **STH:** STH (Zhang et al., 2010) aims to find the best  $l$ -bit binary codes for all documents in the corpus via unsupervised learning.
- **AGH:** This method (Liu et al., 2011) discovers the neighborhood structure hidden in the data to learn proper compact codes. To make the method computationally feasible, it utilizes Anchor Graphs to gain tractable low-rank adjacency matrices.
- **VDSH:** (Chaidaroon and Fang, 2017) presents a series of deep learning models for text hashing, including VDSH, VDSH-S, and VDSH-SP. The VDSH-S and VDSH-SP models are supervised by utilizing document labels/tags for the hashing process. For the comparison's fairness, the VDSH is adopted as the baseline since our method is also unsupervised.

To better evaluate the effectiveness of hash codes used in the field of similarity search, every document in the test set is adopted as the query document. The similarity between the query document

<sup>4</sup><http://ana.cachopo.org/datasets-for-single-label-text-categorization>

and each target similar document, which is utilized to retrieve relevant documents, is calculated by the Hamming distance of their hash codes respectively. The performance is measured by Precision, which is the ratio of the number of the similar documents to the number of total retrieved documents. The retrieved document that shares any common test label with the query document is denoted as a relevant document. Similar to previous works (Chaidaroon and Fang, 2017), the precision for the top 100 (pre@100) is employed as the main criterion. The final results are averaged over all the test documents.

### 3.3 Experimental Setup

We randomly split each dataset into two subsets for training and testing, which account for 90% and 10%, respectively. The training data is used to learn the mapping from the document to the hash code. Each document in the test set is used to retrieve similar documents based on the mapping, and the results are evaluated. The similar as (Chaidaroon and Fang, 2017), we use one-hot encoding as the default representation of the raw document. The hyper-parameter  $\lambda = [0, \mathbf{0.001}, 0.01, 0.1]$ , where the number in bold denotes the default setting (see Section 3.7). The number of iterations is set to 10 (see Section 3.8). Our codes are available at the open-source code repository <sup>5</sup>.

In addition, the settings of the SH<sup>6</sup>, AGH<sup>7</sup>, STH<sup>8</sup> and VDSH<sup>9</sup> remain unchanged with original paper. We run five trials for each methods and an average of five trials is reported to avoid bias introduced by randomness. All of the methods are run on Windows with 1 Intel i7-7500 CPU and 1 GeForce GTX 1050Ti GPU.

### 3.4 Comparison Results

To examine the competitiveness of BCTH, we compared our method with competitive baselines, including traditional techniques and deep learning models from the perspective of both precision and training speed.

Table 2 reports the training time on the 20Newsgroups dataset. From the table, we can derive the following interesting conclusions: (1) Compared

<sup>5</sup><https://github.com/myazi/SemHash>

<sup>6</sup><https://github.com/superhans/SpectralHashing>

<sup>7</sup><https://github.com/ColumbiaDVMM/Anchor-Graph-Hashing>

<sup>8</sup>[http://www.dcs.bbk.ac.uk/~dell/publications/dellzhang\\_sigir2010/sth\\_v1.zip](http://www.dcs.bbk.ac.uk/~dell/publications/dellzhang_sigir2010/sth_v1.zip)

<sup>9</sup><https://github.com/unsuthee/VariationalDeepSemanticHashing>

Table 1: Precision of the top 100 retrieved documents on four datasets with different numbers of hashing bits. The bold font denotes the best result at that number of bits.

Methods	RCV1					Reuters				
	8bits	16bits	32bits	64bits	128bits	8bits	16bits	32bits	64bits	128bits
LSH	0.4180	0.4352	0.4716	0.5214	0.5877	0.2802	0.3215	0.3862	0.4667	0.5194
SH	0.5321	0.5658	0.6786	0.7337	0.7064	0.4016	0.4201	0.4631	0.4590	0.4622
STH	0.6992	0.7688	0.8016	0.8098	0.8037	<b>0.6955</b>	0.7239	0.7576	0.7486	0.7240
AGH	0.4257	0.4976	0.5457	0.5698	0.5799	0.6552	0.7046	0.7313	0.7189	0.7043
VDSH	0.7285	0.7718	0.8165	0.7720	0.6630	0.6642	0.7118	0.7335	0.7083	0.7079
BCTH	<b>0.7339</b>	<b>0.7989</b>	<b>0.8389</b>	<b>0.8641</b>	<b>0.8690</b>	0.6827	<b>0.7307</b>	<b>0.7584</b>	<b>0.7669</b>	<b>0.7889</b>
Methods	20Newsgroups					TMC				
	8bits	16bits	32bits	64bits	128bits	8bits	16bits	32bits	64bits	128bits
LSH	0.0578	0.0597	0.0666	0.0770	0.0949	0.4388	0.4393	0.4514	0.4553	0.4773
SH	0.0699	0.1096	0.2010	0.2732	0.2632	0.5999	0.6206	0.6108	0.5813	0.5612
STH	0.2035	0.3481	0.4581	0.5129	0.5247	<b>0.7278</b>	<b>0.7520</b>	0.7633	0.7569	0.7411
AGH	0.2435	0.3531	0.3861	0.3796	0.3579	0.6000	0.6334	0.6443	0.6423	0.6273
VDSH	<b>0.3514</b>	0.3848	0.4667	0.2219	0.0651	0.6503	0.6640	0.7062	0.6567	0.5868
BCTH	0.3089	<b>0.4497</b>	<b>0.5216</b>	<b>0.5534</b>	<b>0.5830</b>	0.7076	0.7351	<b>0.7651</b>	<b>0.7804</b>	<b>0.7926</b>

Methods	8bits	16bits	32bits	64bits	128bits
SH	28.1	28.9	32.2	37.8	47.1
STH	16.3	16.5	17.9	20.3	28.3
AGH	10.3	10.8	11.4	12.8	15.5
VDSH	100+	100+	100+	100+	100+
BCTH	<b>0.5</b>	<b>0.9</b>	<b>2.0</b>	<b>5.0</b>	<b>10.8</b>

Table 2: Training time (second) of different methods on *20Newsgroups* dataset.

with these methods, BCTH costs less training time among all different hash bits. The reason can be attributed to the joint learning of hash codes and hash function, without needing to build the pairwise similarity matrix and the linear time complexity of BTCH. (2) It consumes extremely more time to train the VDSH model than train a traditional model. It shows that deep learning methods with sophisticated network architecture bring many parameters, thus requiring much more time to complete the training process. (3) The SH, STH, and AGH spend less time on the training process, which indicates that the traditional methods has its advantage over the deep learning method in training time.

Apart from comparing the 20Newsgroups dataset, we also compare over four datasets from the perspective of precision. Table 1 reports the comparison results with various methods over different numbers of bits. From this table, we can derive the following interesting conclusions: (1) Our proposed BCTH outperforms nearly all baselines among all different hash bits on four datasets.

It demonstrates that BCTH, which introduces the bit-balanced and the bit-uncorrelated constraints, can learn effectively hash code from documents. (2) The VDSH model outperforms traditional methods in almost all situations. It denotes that deep learning techniques can capture inherent hidden text semantics, which are beneficial to generate the text hash codes. Although our method does not get the best results in some datasets under the circumstance of short hashing bit, it is approximating the best ones. Since our method utilizes the bit-balanced and the bit-uncorrelated constraints to make each bit capture independent semantics for documents, it is worthwhile to study the relationship between the length of the hash codes and the effect of our method.

### 3.5 Impact of the Length of Hash Codes

Previous works usually limit the length of the hash code to 128 bits on account of data storage. To study the effectiveness of the hash codes' size, we conduct experiments on hash codes ranging from 8 bits to 128 bits and extend hash codes to 1024 bits in this section.

Figure 2 reports the compared results on four datasets. From this figure, we can find the following phenomena: (1) when the length of the hash codes is equal to or over 128 bits, the effect of most other methods starts to decline. (2) the performance of our method always increases with the length of the hash codes increasing over all datasets. The

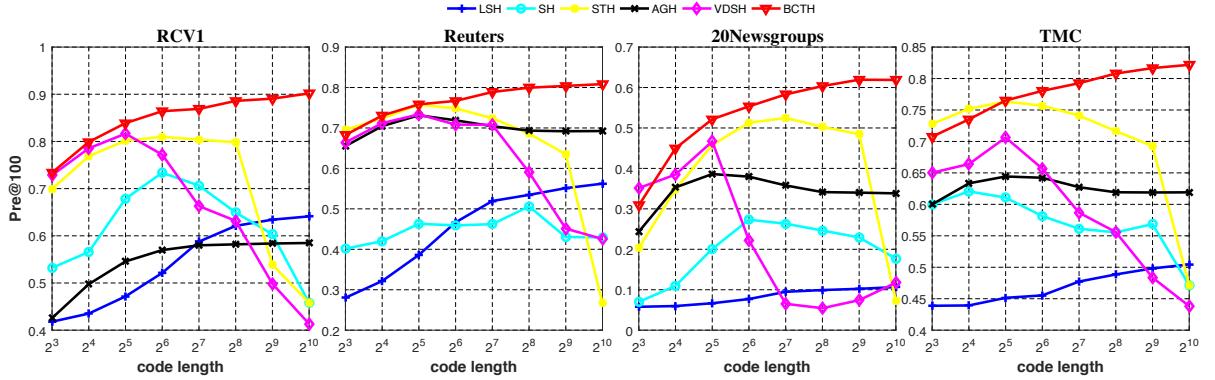


Figure 2: Precision@100 curve on four datasets with hash codes length from 8 to 1024.

reason is that our approach, which introduces the bit-balanced and the bit-uncorrelated constraints, can better keep independent semantic for all bits.

### 3.6 Qualitative Analysis

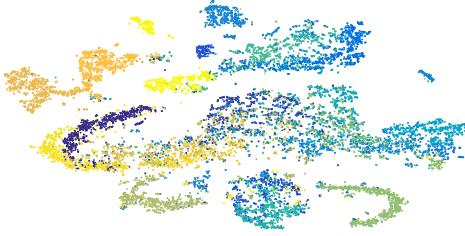


Figure 3: Visualization of the 1024-dimensional document latent semantic vectors by BCTH on the 20Newsgroup dataset using t-SNE.

To evaluate whether our presented BCTH model can preserve the original documents’ semantics, we visualize the documents’ low-dimensional representations on the 20Newsgroups dataset in this section. In particular, the hash codes, obtained by BCTH, can be regarded as the latent semantic vectors of documents. We use t-SNE<sup>10</sup> tool to generate the scatter plots through 1024-bit hash codes. Figure 3 shows the results. Different colors represent different categories based on the ground truth. As we can see from figure 3, BCTH generates well-separated clusters with each corresponding to a true category. It shows that our method can effectively learn low-dimensional representations for documents.

### 3.7 Impact of Parameters

Our method is involved with a critical parameter  $\lambda$ , which is used to control the bit-balanced and the bit-

Datasets	$\lambda = 0$	$\lambda = 0.001$	$\lambda = 0.01$	$\lambda = 0.1$
RCV1	0.8476	<b>0.8641</b>	0.8526	0.8269
Reuters	0.7577	<b>0.7669</b>	0.7668	0.7532
20Newsgroups	0.5528	<b>0.5534</b>	0.5464	0.5502
TMC	0.7073	<b>0.7172</b>	0.7070	0.7025

Table 3: The effect of  $\lambda$  on four datasets with 64 hashing bits.

uncorrelated constraints. We here study the impact of hyper-parameter  $\lambda$  in this section. Table 3 shows the results, which are obtained by using 64 hash bits. From this table, we can find that: (1) With  $\lambda$  varying from 0 to 0.1, BCTH is able to achieve relatively desirable results over all four datasets, which means that  $\lambda$  is universally applicable; (2) With  $\lambda$  set to 0.001, BCTH obtains the optimal result, and therefore, 0.001 is set as the default value for our method.

### 3.8 Convergence Speed

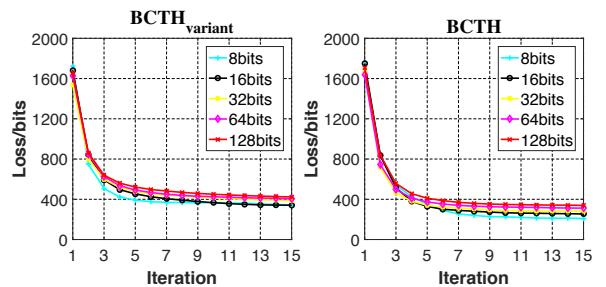


Figure 4: Convergence curve of the loss on the 20Newsgroups.

In order to evaluate the convergence performance of our proposed BCTH algorithm, we performed convergence experiments on the 20Newsgroups dataset. Considering the different loss scales produced under different hash bits, we consider the ratio of the loss to the hash length to make

<sup>10</sup><https://lvdmaaten.github.io/tsne/>

it comparable at the same scale for different iterations. Note that, in this set of experiments, we also test one variant of BCTH methods, which is known as BCTH<sub>variant</sub>, which calculates the conditional probability through Eq. (2). The result is reported in figure 4, and we can find that: (1) both the BCTH and the BCTH<sub>variant</sub> converge after approximately 10 iterations, and therefore, 10 is set as the default value for our method; (2) the convergence effect of BCTH is better than BCTH<sub>variant</sub>, which has a lower loss. This demonstrates that BTCH is effective.

## 4 Related Work

Hashing methods can be divided into data-independent methods and data-dependent methods (Chang et al., 2012). The well-known data-independent methods include locality sensitive hashing (LSH) (Datar et al., 2004) and its variants. Data-dependent hashing methods are also known as learning to hash (L2H) methods by learning a hash function from data (Li et al., 2016). At present, the main L2H methods (Wang et al., 2018) can be divided into three categories: pairwise similarity preserving, multiwise similarity preserving, and implicit similarity preserving. The pairwise similarity-preserving methods aim to build a pairwise similarity matrix between two points, such as spectral hashing (SH) (Weiss et al., 2008), hashing with graphs (AGH) (Liu et al., 2011), discrete graph hashing (DGH) (Liu et al., 2014), fast supervised hashing (FastH) (Lin et al., 2014a) and column-sampling-based discrete supervised hashing (COSDISH) (Kang et al., 2016). The multiwise similarity-preserving is similar to pairwise similarity, which uses three or more samples as a group to define generalized similarity measures (Norouzi et al., 2012; Wang et al., 2013a). The implicit similarity-preserving methods maintain the similarity in an equivalent manner that adopts the idea of preserving the similarity of local neighbors (Irie et al., 2014). Compared with this line of works, although our work also focuses on the nearest neighbor search, our work is different from theirs since (1) most of these works focus on images data, and (2) Bayesian Clustering is not covered in these works.

Another line of works discuss text hashing, is related to our work since our work also aims to learn binary code from documents effectively. For example, (Zhang et al., 2010) presented the Self-Taught

Hashing (STH) method for efficiently learning semantic hashing. (Zhang et al., 2010) incorporated both the tag information and the similarity information from probabilistic topic modeling. However, many of these models rely on pairwise similarity-preserving technique, which the time complexity is unavoidable  $O(m^2)$  where m is the number of documents. On the other hand, researchers have attempted to study text hashing (Xu et al., 2015) via deep neural networks owing to the success of deep learning. For example, (Chaidaroon and Fang, 2017) introduces a latent factor for documents to capture the semantic information. (Kalchbrenner et al., 2014) proposed an end-to-end Neural Architecture for Semantic Hashing (NASH), which treats the hashing codes as latent variables. Compared to this line of works, our work shares several common features: (1) our work also learns hashing by introducing latent factor, and (2) our work also aims to the issues related to text hashing. Nevertheless, our work differs from theirs in several features: (1) most of these works are based on complex nonlinear functions like convolutional neural networks, and training time complexity is enormous, and (2) Bayesian Clustering is not covered in these works. In this paper, we make the first attempts to utilize Bayesian Clustering for text hashing and gain training time's linear complexity.

## 5 Conclusion

This paper presents a general learning framework that utilizes multiple Bayesian Clusterings jointly for text hashing. We introduce two constraints to make the hash code effectively. Specifically, the bit-balanced constraint is employed to maximize the amount of information in each bit, and the bit-uncorrelated constraint is adopted to keep the independence among all bits. The time complexity of our method is linear. Based on four widely-used datasets, the experiment results demonstrate that BCTH is competitive compared with current competitive baselines from the perspective of both precision and training speed.

## References

- Alexandr Andoni and Piotr Indyk. 2006. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *IEEE Symposium on Foundations of Computer Science*, pages 459–468.
- Suthee Chaidaroon and Yi Fang. 2017. Variational deep semantic hashing for text documents. In *Pro-*

- ceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 75–84.
- ShihFu Chang, YuGang Jiang, Rongrong Ji, Jun Wang, and Wei Liu. 2012. Supervised hashing with kernels. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2074–2081.
- Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*.
- Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab S. Mirrokni. 2004. Locality-sensitive hashing scheme based on p-stable distributions. In *Twenty-sixth Symposium on Computational Geometry*, pages 253–262.
- Aristides Gionis, Piotr Indyk, and Rajeev Motwani. 1999. Similarity search in high dimensions via hashing. In *Proceedings of Vldb Conference*, volume 8, pages 518–529.
- Go Irie, Zhenguo Li, Xiao Ming Wu, and Shih Fu Chang. 2014. Locally linear hashing for extracting non-linear manifolds. In *Computer Vision and Pattern Recognition*, pages 2123–2130.
- Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. In *Proceedings of the Association for Computational Linguistics*.
- Wang Cheng Kang, Wu Jun Li, and Zhi Hua Zhou. 2016. Column sampling based discrete supervised hashing. In *AAAI Conference on Artificial Intelligence*, pages 1230–1236.
- Yuquan Le, Zhi-Jie Wang, Zhe Quan, Jiawei He, and Bin Yao. 2018. Acv-tree: A new method for sentence similarity modeling. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, pages 4137–4143.
- Wu Jun Li, Sheng Wang, and Wang Cheng Kang. 2016. Feature learning based deep supervised hashing with pairwise labels. In *International Joint Conference on Artificial Intelligence*, pages 1711–1717.
- Guosheng Lin, Chunhua Shen, Qinfeng Shi, Anton Van Den Hengel, and David Suter. 2014a. Fast supervised hashing with decision trees for high-dimensional data. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1971–1978.
- Guosheng Lin, Chunhua Shen, David Suter, and Anton Van Den Hengel. 2014b. A general two-step approach to learning-based hashing. In *IEEE International Conference on Computer Vision*, pages 2552–2559.
- W. Liu, C. Mu, S. Kumar, and S. F. Chang. 2014. Discrete graph hashing. *Advances in Neural Information Processing Systems*, 4:3419–3427.
- Wei Liu, Jun Wang, Sanjiv Kumar, and Shih Fu Chang. 2011. Hashing with graphs. In *Proc. International Conference on Machine Learning June*, pages 1–8.
- Mohammad Norouzi, David J Fleet, and Ruslan Salakhutdinov. 2012. Hamming distance metric learning. *Advances in Neural Information Processing Systems*, 2:1061–1069.
- Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to rank short text pairs with convolutional deep neural networks. In *The International ACM SIGIR Conference*, pages 373–382.
- J. Wang, T. Zhang, J. Song, N. Sebe, and H. T. Shen. 2018. A survey on learning to hash. *IEEE Transactions on Pattern Analysis Machine Intelligence*, PP(99):769–790.
- Jianfeng Wang, Jingdong Wang, Nenghai Yu, and Shipeng Li. 2013a. Order preserving hashing for approximate nearest neighbor search. In *Proceedings of the 21st ACM international conference on Multimedia*, pages 133–142.
- Jun Wang, Wei Liu, Sanjiv Kumar, and Shih Fu Chang. 2016. Learning to hash for indexing big data - a survey. *Proceedings of the IEEE*, 104(1):34–57.
- Qifan Wang, Dan Zhang, and Luo Si. 2013b. Semantic hashing using tags and topic modeling. In *International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 213–222.
- Yair Weiss, Antonio Torralba, and Robert Fergus. 2008. Spectral hashing. In *International Conference on Neural Information Processing Systems*, pages 1753–1760.
- Jiaming Xu, Peng Wang, Guanhua Tian, Bo Xu, Jun Zhao, Fangyuan Wang, and Hongwei Hao. 2015. Convolutional neural networks for text hashing. In *International Conference on Artificial Intelligence*, pages 1369–1375.
- Dell Zhang, Jun Wang, Deng Cai, and Jinsong Lu. 2010. Self-taught hashing for fast similarity search. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 18–25.
- Hanwang Zhang, Fumin Shen, Wei Liu, Xiangnan He, Huanbo Luan, and Tat Seng Chua. 2016. Discrete collaborative filtering. In *International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 325–334.

# Lightweight Text Classifier using Sinusoidal Positional Encoding

Byoung-Doo Oh and Yu-Seop Kim

Department of Convergence Software, Hallym University, Republic of Korea

iambd822@gmail.com, yskim01@hallym.ac.kr

## Abstract

Large and complex models have recently been developed that require many parameters and much time to solve various problems in natural language processing. This paper explores an efficient way to avoid models being too complicated and ensure nearly equal performance to models showing the state-of-the-art. We propose a single convolutional neural network (CNN) using the sinusoidal positional encoding (SPE) in text classification. The SPE provides useful position information of a word and can construct a more efficient model architecture than before in a CNN-based approach. Our model can significantly reduce the parameter size (at least 67%) and training time (up to 85%) while maintaining similar performance to the CNN-based approach on multiple benchmark datasets.

## 1 Introduction

In recent years, convolutional neural networks (CNN) have shown remarkable performance and time-efficiency in text classification tasks such as sentiment analysis and document classification. (Kim, 2014; Kalchbrenner et al., 2014; Zhang et al., 2015). In particular, Kim (2014) shows the importance of pre-trained word vectors and fine-tuned word vectors for each task, currently being used in many studies. However, in CNN, convolution and pooling operations lose information about the local order of words (Britz, 2015; Yenigalla et al., 2018). To solve this problem, various studies have been conducted on model architectures capable of effectively extracting features from CNN (Kim, 2014; Kalchbrenner et al., 2014; Zhang et al., 2015; Lai et al., 2015; Zhao et al., 2018; Kim et al., 2020).

On the other hand, when using a word vector based on a distributed representation, fine-tuning of the pre-trained word vector for each task requires more parameters in proportion to the sequence length. Along with this, the proposed large and complex model architectures have improved performance, but additional space and time costs are required due to numerous parameters (Alom et al., 2019). Recently, studies have been conducted on efficient model architectures with less parameter size and computational cost without significant performance loss compared to existing models (Vaswani et al., 2017; Tay et al., 2019; Zhang and Sennrich, 2019).

Vaswani et al. (2017) introduced sinusoidal positional encoding (SPE) and multi-head self-attention to introduce a simple model architecture with less parameter size and computational cost than before. The SPE provides useful word position information without recurrent neural networks (RNN) and, at the same time, requires less parameter size. Due to these advantages, studies have been conducted to show the usefulness of location encoding in computer vision and natural language processing (Takase and Okazaki, 2019; Islam et al., 2020).

In this paper, we propose a single CNN with SPE and construct a simpler model architecture than before with useful position information. SPE's position information is applied to the pre-trained word vector, and it is maintained as a static vector without fine-tuning. A single CNN extracts significant features from word vectors containing position information. The results of this study are confirmed against six benchmark datasets. The proposed model is challenging to achieve state-of-the-art, but it maintained similar performance as before, despite significantly reducing the parameter size and time cost compared to the previous CNN-based approaches.

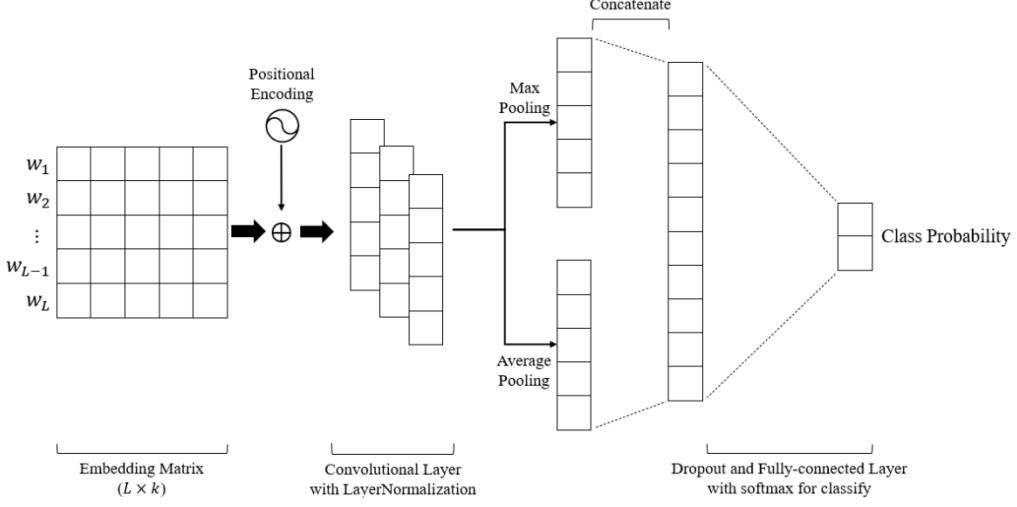


Figure 1: Model architecture of Single CNN with Sinusoidal Positional Encoding.

This paper is organized as follows. Section 2 describes the model proposed in this paper. Section 3 describes hyper-parameters and experimental environments, and Section 4 describes the evaluation and the experimental results. Finally, Section 5 concludes with a summary of what we have identified.

## 2 Model

As shown in Figure 1, the proposed model's architecture is based on Collobert et al. (2011). The proposed structure is slightly modified. In this paper, a model consisting of SPE and a single CNN is proposed. A single CNN consists of three layers: convolutional layer, pooling layer (max and average), and fully-connected layer with softmax. Each component is described in detail in the rest of this section.

### 2.1 Sinusoidal Positional Encoding

CNN is difficult to learn word order in sentences (Britz, 2015; Yenigalla et al., 2018). For example, CNN learns “The wolves ate” and “ate the wolves” as the same representation. Therefore, studies have been conducted to effectively provide sequential information to neural network models, excluding RNN (Yang et al., 2016; Gehring et al., 2017; Vaswani et al., 2017).

The SPE introduced by Vaswani et al. (2017) uses sine and cosine functions to represent each word's relative position in an embedding. Besides, it provides useful position information with a parameter-free position representation. SPE ( $PE_{(pos,i)}$ ) is calculated as follows:

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/dim}}\right) \quad (1)$$

$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/dim}}\right) \quad (2)$$

, where  $pos$  is the position of each word in the embedding,  $i$  is the position of the dimension in the word vector, and  $dim$  is the size of the word vector. Therefore, SPE provides the position information by calculating the word vector's even and odd dimensions with sine and cosine functions, respectively.

In this paper, each word's position information obtained from SPE ( $\vec{p}_t$ ) was added to the word vector ( $x_i$ ), as shown in Equation (3).

$$\vec{x}_i = x_i + \vec{p}_t \quad (3)$$

The SPE's position information provides useful information when a single CNN extracts features from a word vector. Through this, fine-tuning for word vectors is not considered in training. This can reduce the parameter size required for fine-tuning. The word vector generated in this way is transferred to a single CNN.

### 2.2 Single Convolutional Neural Networks

$x_i \in \mathbb{R}^k$  is a  $k$ -dimensional word vector constituting a sentence, and if the length of the sentence is  $L$ , the embedding matrix is represented as  $x_{i:L} \in \mathbb{R}^{L \times k}$ . The weight of the convolution filter applied to the embedding matrix is represented as  $w \in \mathbb{R}^{j \times k}$ , and a new feature  $c_i$  is generated from  $j$  word vectors represented in  $k$ -dimensions. For example, feature  $c_i$  (when stride=1) is created as follows:

Dataset	$c$	$D_{train}$	$D_{val}$	$D_{test}$	$ V $	$ V_{pre} $
IMDB	2	22,500	2,500	25,000	112,540	58,843
MR	2	8,635	960	1,067	18,764	16,448
MPQA	2	8,587	955	1,067	6,246	6,083
TREC	6	4,843	539	500	8,689	7,461
Reuters	10	6,472	720	2,787	28,482	17,508
20news	20	10,182	1,132	7,532	117,925	50,021

Table 1: Summary for the datasets after tokenization.  $c$ : Number of class.  $D_{train}$ : train size.  $D_{val}$ : validation size.  $D_{test}$ : test size.  $|V|$ : Vocabulary size.  $|V_{pre}|$ : Number of words presented in the set of pre-trained word vectors.

$$c_i = f(w \cdot x_{i:i+j-1} + b) \quad (4)$$

In Equation (4),  $f$  is the activation function, and  $b \in \mathbb{R}$  is the bias value. This convolution filter is applied to all possible  $j$  words in a sentence, and all features are concatenated to generate a single feature map  $c$ .

$$c = [c_1, c_2, c_3, \dots, c_{i-j+1}] \quad (5)$$

In this paper, the feature map  $c$  obtained from the convolutional layer is transferred after normalization. The normalized feature map  $c$  applies a max-pooling layer ( $\hat{c} = \max(c)$ ) that extracts the maximum value and an average-pooling layer ( $\tilde{c} = \text{avg}(c)$ ) that extracts the average value. The obtained features are concatenated and transferred to a fully-connected layer with softmax after regularization to classify the class.

### 2.3 Normalization and Regularization

We used LayerNormalization (Ba et al., 2016) for feature map  $c$ . LayerNormalization is applied independently to each feature map for normalization. Furthermore, since it is normalized to the mean ( $\mu$ ) and variance ( $\sigma$ ), it can be applied equally to training and test data and has high time-efficiency.

$$\mu^l = \frac{1}{H} \sum_{i=1}^H a_i^l \quad (6)$$

$$\sigma^l = \sqrt{\frac{1}{H} \sum_{i=1}^H (a_i^l - \mu_i^l)^2} \quad (7)$$

, where  $H$  is the number of hidden nodes in the layer and  $a_i$  is the  $i$ -th vector at the hidden nodes.

For regularization, we used Dropout (Srivastava et al., 2014) for feature  $z = [\hat{c}_1, \dots, \hat{c}_m, \tilde{c}_1, \dots, \tilde{c}_m]$

obtained from the max-pooling layer and the average-pooling layer.

## 3 Experimental Setup

### 3.1 Datasets

In this paper, six benchmark datasets - Sentiment classification: Internet Movie Database (IMDB)<sup>1</sup>, movie review dataset (MR)<sup>2</sup> (Pang and Lee, 2005). Question categorization: MPQA dataset<sup>3</sup> (Wiebe et al., 2005), TREC question dataset<sup>4</sup> (Li and Roth, 2002). News categorization: Reuters dataset<sup>5</sup> and 20news dataset<sup>6</sup> - are used. Table 1 summarizes each dataset.

### 3.2 Implementation Details

In this paper, the GloVe<sup>7</sup> (Pennington et al., 2014) vector, which pre-trained with 840 billion words provided by Stanford University, is used, representing the word vector as 300 dimensions.

The hyper-parameters are set the same for all datasets. The activate function used in the convolution layer is a linear function, not a nonlinear function such as ReLU (Nair and Hinton, 2010) or hyperbolic tangent. The filter window size ( $j$ ) is 3, the number of filters is 128, the l2\_regularizer is 0.0001, the epsilon in LayerNormalization is 1e-6, the dropout rate is 0.1, and the mini-batch size is 40. These values were determined experimentally.

Also, we use only early stops when training. As shown in Table 1, the validation dataset is randomly selected, and it is 10% of the training dataset. Furthermore, optimization is performed using Adam optimizer (Kingma and Ba, 2015), and the learning ratio is set to 0.0001.

<sup>1</sup> <https://ai.stanford.edu/~amaas/data/sentiment/>

<sup>2</sup> <https://www.cs.cornell.edu/people/pabo/movie-review-data/>

<sup>3</sup> <https://mpqa.cs.pitt.edu/>

<sup>4</sup> <https://cogcomp.seas.upenn.edu/Data/QA/QC/>

<sup>5</sup> <https://www.nltk.org/book/ch02.html> (used only the data of the 10 largest topics)

<sup>6</sup> [https://scikit-learn.org/0.19/datasets/twenty\\_newsgroups.html](https://scikit-learn.org/0.19/datasets/twenty_newsgroups.html)

<sup>7</sup> <https://nlp.stanford.edu/projects/glove/>

Model	Type	IMDB	MR	MPQA	TREC	Reuters	20news
CNN-static	P	361K	361K	361K	362K	363K	366K
	T	40	<b>15</b>	<b>15</b>	<b>8</b>	12	18
CNN-nonstatic	P	37,637K	6,208K	2,245K	2,969K	9,260K	54,130K
	T	91	18	16	9	15	52
DCNN	P	37,874K	6,445K	2,482K	3,205K	9,497K	54,366K
	T	146	38	36	21	32	77
Capsule-A	P	37,318K	5,889K	1,926K	2,649K	8,941K	53,810K
	T	230	32	30	17	60	111
Capsule-B	P	37,431K	6,002K	2,039K	2,764K	9,058K	53,932K
	T	425	67	65	35	120	213
CapsNet-static	P	47,970K	9,354K	3,158K	7,227K	21,627K	66,503K
	T	287	37	31	21	76	159
CapsNet-Dynamic	P	47,970K	9,354K	3,158K	7,227K	21,627K	66,503K
	T	289	37	31	22	76	160
SingleCNN-static	P	116K	116K	116K	117K	118K	121K
	T	36	15	15	8	10	17
SingleCNN-nonstatic	P	37,392K	5,963K	53,884K	9,015K	2,000K	2,724K
	T	85	18	16	9	13	50
Single CNN-SPE (Proposed Model)	P	<b>116K</b>	<b>116K</b>	<b>116K</b>	<b>117K</b>	<b>118K</b>	<b>121K</b>
	T	<b>36</b>	<b>15</b>	<b>15</b>	<b>8</b>	<b>10</b>	<b>17</b>

Table 2: Experimental results of parameter size and training times (P: parameter size, T: seconds/epoch). The best values in each dataset is shown in bolded.

Model	IMDB	MR	MPQA	TREC	Reuters	20news
CNN-static	89.72	79.38	87.73	90.84	86.83	83.18
CNN-nonstatic	89.69	<b>81.09</b>	88.42	91.12	87.17	<b>84.63</b>
DCNN	89.6	79.53	89.26	89.32	87.06	81.55
Capsule-A	86.16	78.22	86.93	82.94	82.29	60.65
Capsule-B	89	79.74	88.19	82.95	<b>88.1</b>	69.97
CapsNet-static	87.18	77.96	88.75	92.08	87.58	81.34
CapsNet-Dynamic	86.47	78.31	<b>89.2</b>	92.35	87.45	82.42
SingleCNN-static	90.37	80.23	88.43	91.08	86.03	81.71
SingleCNN-nonstatic	<b>90.58</b>	80.02	88.41	91.13	86.91	83.78
SingleCNN-SPE (Proposed Model)	90.44	80.93	88.88	<b>92.52</b>	86.89	82.95

Table 3: Experimental results of accuracy. Models marked with an asterisk used the code published by the author on GitHub for accurate implementation.

### 3.3 Baseline Model

In this paper, to confirm our model's performance, we compare it with various models using only CNN as follows: CNN-static and CNN-nonstatic (Kim, 2014), DCNN (Kalchbrenner et al., 2014), Capsule-A and Capsule-B (Zhao et al., 2018), CapsNet-static and CapsNet-dynamic (Kim et al., 2020). These models are set identically to the hyper-parameters proposed in each paper.

We also identify a model that does not use SPE to confirm SPE's usefulness in the CNN-based approach and are as follows: SingleSCNN-static (without fine-tuning) and SingleSCNN-nonstatic (with fine-tuning).

## 4 Results and Discussion

First, we check the parameter size and training time. When measuring the training time, all models set the mini-batch size to 5. The results are shown in Table 2.

Each model, except the proposed model, performs fine-tuning in training. The size of the parameters, including this, is shown in Table 2. In this case, our model reduces the parameter size significantly, which is compared to the previous one. Even in excluding fine-tuning, the parameter size could be reduced by about 67% (CNN-static).

The training times are reduced by up to 90% based on the IMDB with the large-scale dataset,

Model	MR	MPQA	TREC	Reuters
SingleCNN-static	71.68	82.86	83.76	84.91
SingleCNN-SPE	52.26	67.77	86.6	83.76

Table 4: Experimental results using simple bag-of-words.

and only CNN-static shows a similar training time to our model.

Table 3 shows the performance of the models composed of the parameter size in Table 2. In the performance evaluation, all methods are repeated five times, and the average accuracy is measured. Our model maintains similar performance to other models, despite reducing the parameter size by at least 67%.

Additionally, we confirmed SPE's usefulness for word vectors represented based on simple bag-of-words, as shown in Table 4. Originally, SPE calculates each word's position information from the word vector based on the distributed representation. Therefore, as shown in Table 3, when the word vectors based on distributed representation was used, SPE's usefulness was confirmed. Experimental results with bag-of-words based word vectors were not good. It was also confirmed that the SPE's position information is difficult to provide efficient information in the bag-of-words based word vector.

#### 4.1 SPE vs Fine-tuning

Kim (2014) confirmed the usefulness of fine-tuning for pre-trained word vectors, and it is still used in many research. However, this process requires additional parameters in addition to the parameters of the model in training. Therefore, we compared the performance with SingleCNN-static and SingleCNN-nonstatic to confirm SPE's usefulness in the CNN-based approach.

As shown in Table 3, the position information obtained by the SPE confirmed a result similar to the effect of fine-tuning in the CNN-based approach. It is considered that the position information obtained by the SPE can have an efficient effect without fine-tuning in the CNN-based approach.

Additionally, other models' performance was different from that of the previous paper, which is considered to have arisen from the experimental environment's difference.

## 5 Conclusion

In this paper, we propose a single convolutional neural network to which sinusoidal positional encoding is applied. Besides, we describe an experiment to confirm the usefulness of sinusoidal positional encoding in the CNN-based approach. This model shows excellent performance despite being lightweight in text classification. Although this model reduced the parameter size by at least 67%, it was possible to confirm a similar performance to the previous model. Besides, a similar effect could be confirmed without fine-tuning in the CNN-based approach.

In the future, we intend to study a method that can easily learn the position information of words in a manner other than SPE. Through this, we want to develop a more straightforward and more powerful model that can show both the time-advantageous CNN and the sequential learning power of RNN.

## Acknowledgement

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2019R1A2C2006010).

## Reference

- Md Zahangir Alom, Tarek M. Taha, Chris Yakopcic, Stefan Westberg, Paheding Sidike, Mst Shamima Nasrin, Mahmudul Hasan, Brian C. Van Essen, Abdul A. S. Awwal, and Vijayan K. Asari. 2019. A state-of-the-art survey on deep learning theory and architectures. *Electronics*, 8(3):292. <https://doi.org/10.3390/electronics8030292>.
- Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer Normalization. arXiv preprint arXiv:1607.06450.
- Denny Britz. 2015. Understanding convolutional neural networks for NLP. <http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/>.
- Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural Language Processing (Almost) from

- Scratch. Journal of machine learning research, 12:2493-2537.
- Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional Sequence to Sequence Learning. In Proceedings of the 34th International Conference on Machine Learning (Volume 70), pages 1243-1252.
- Md Amirul Islam, Sen Jia, and Neil D. B. Bruce. 2020. How Much Position Information Do Convolutional Neural Networks Encode?. In Proceedings of the 8th International Conference on Learning Representations.
- Nal Kalchbrenner, Edward Grefenstette, and Phill Blunsom. 2014. A convolutional neural network for modelling sentences. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, pages 655-665. <https://www.aclweb.org/anthology/P14-1062>.
- Jaeyoung Kim, Sion Jang, Eunjeong Park, and Sungchul Choi. 2020. Text classification using capsules. *Neurocomputing*, 376:214-221. <https://doi.org/10.1016/j.neucom.2019.10.033>.
- Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, pages 1746-1751. <https://www.aclweb.org/anthology/D14-1181>.
- Diederik P. Kingma and Jimmy Lei Ba. 2015. Adam: A Method for Stochastic Optimization. In Proceedings of the 3rd International Conference on Learning Representations.
- Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent Convolutional Neural Networks for Text Classification. In Proceedings of the Twenty-ninth AAAI Conference on Artificial Intelligence, pages 2267-2273.
- Xin Li and Dan Roth. 2002. Learning Question Classifiers. In Proceeding of the 19th International Conference on Computational Linguistics. <https://www.aclweb.org/anthology/C02-1150>.
- Vinod Nair and Geoffrey E. Hinton. 2010. Rectified Linear Units Improve Restricted Boltzmann Machines. In Proceedings of the 27th International Conference on Machine Learning, pages 807-814.
- Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, pages 115-124. <https://www.aclweb.org/anthology/P05-1015>.
- Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global Vectors for Word Representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, pages 1532-1543. <https://www.aclweb.org/anthology/D14-1162>.
- Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of machine learning research*, 15(1):1929-1958.
- Sho Takase and Naoaki Okazaki. 2019. Positional Encoding to Control Output Sequence Length. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 3999-4004. <https://www.aclweb.org/anthology/N19-1401>.
- Yi Tay, Aston Zhang, Luu Anh Tuan, Jinfeng Rao, Shuai Zhang, Shuohang Wang, Jie Fu, and Siu Cheung Hui. 2019. Lightweight and Efficient Neural Natural Language Processing with Quaternion Networks. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 1494-1503. <https://www.aclweb.org/anthology/P19-1145>.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In Advances in Neural Information Processing Systems, pages 5998-6008.
- Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating Expressions of Opinions and Emotions in Language. *Language Resources and Evaluation*, 39(2-3):165-210. <https://doi.org/10.1007/s10579-005-7880-9>.
- Yunlun Yang, Yunhai Tong, Shulei Ma, and Zhi-Hong Deng. 2016. A Position Encoding Convolutional Neural Network Based on Dependency Tree for Relation Classification. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 65-74. <https://www.aclweb.org/anthology/D16-1007>.
- Promod Yenigalla, Sibsambhu Kar, Chirag Singh, Ajay Nagar, and Gaurav Mathur. 2018. Addressing Unseen Word Problem in Text Classification. In Proceedings of the International Conference on Applications of Natural Language to Information Systems, pages 339-351. [https://doi.org/10.1007/978-3-319-91947-8\\_36](https://doi.org/10.1007/978-3-319-91947-8_36).
- Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level Convolutional Networks for Text Classification. In Advances in Neural Information Processing Systems, pages 649-657.
- Biao Zhang and Rico Sennrich. 2019. A Lightweight Recurrent Networks for Sequence Modeling. In

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 1538-1548. <https://www.aclweb.org/anthology/P19-1149>.

Wei Zhao, Jianbo Ye, Min Yang, Zeyang Lei, Soufei Zhang, and Zhou Zhao. 2018. Investigating Capsule Networks with Dynamic Routing for Text Classification. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 3110-3119. <https://www.aclweb.org/anthology/D18-1350>.

# Towards Non-task-specific Distillation of BERT via Sentence Representation Approximation

Bowen Wu<sup>1,2</sup>, Huan Zhang<sup>1\*</sup>, Mengyuan Li<sup>1\*</sup>, Zongsheng Wang<sup>2</sup>, Qihang Feng<sup>2</sup>,  
Junhong Huang<sup>2</sup>, Baoxun Wang<sup>2</sup>

<sup>1</sup>Peking University, Beijing, China

<sup>2</sup>Platform and Content Group, Tencent

jason\_wbw, zhanghuan123, limengyuan@pku.edu.cn

jasoawang, careyfeng, vincenthuang, asulewang@tencent.com

## Abstract

Recently, BERT has become an essential ingredient of various NLP deep models due to its effectiveness and universal-usability. However, the online deployment of BERT is often blocked by its large-scale parameters and high computational cost. There are plenty of studies showing that the knowledge distillation is efficient in transferring the knowledge from BERT into the model with a smaller size of parameters. Nevertheless, current BERT distillation approaches mainly focus on task-specified distillation, such methodologies lead to the loss of the general semantic knowledge of BERT for universal-usability. In this paper, we propose a sentence representation approximating oriented distillation framework that can distill the pre-trained BERT into a simple LSTM based model without specifying tasks. Consistent with BERT, our distilled model is able to perform transfer learning via fine-tuning to adapt to any sentence-level downstream task. Besides, our model can further cooperate with task-specific distillation procedures. The experimental results on multiple NLP tasks from the GLUE benchmark show that our approach outperforms other task-specific distillation methods or even much larger models, i.e., ELMO, with efficiency well-improved.

## 1 Introduction

As one of the most important progress in the Natural Language Processing field recently, the Bidirectional Encoder Representation from Transformers (BERT) (Devlin et al., 2019) has been proved to be effective in improving the performances of various NLP tasks by providing a powerful pre-trained language model based on large-scale unlabeled corpora. Recent studies have shown that BERT’s capability can be further enhanced by utilizing deeper architectures or performing the pre-training on larger

corpora with appropriate guidance (Radford et al., 2019; Yang et al., 2019; Liu et al., 2019b).

Despite its strength in building distributed semantic representations of sentences and supporting various NLP tasks, BERT holds a huge amount of parameters that raises the difficulty of conducting online deployment due to its unsatisfying computational efficiency. To address this issue, various studies have been done to utilize the knowledge distillation (Hinton et al., 2015) for compressing BERT and meanwhile keep its semantic modeling capability as much as possible (Chia et al., 2019; Tsai et al., 2019). The distilling methodologies include simulating BERT with a much smaller model (e.g., LSTM) (Tang et al., 2019b) and reducing some of the components, such as transformers, attentions to obtain the smaller BERT based model (Sun et al., 2019; Barkan et al., 2019).

Nevertheless, the current methods highly rely on a labeled dataset upon a specified task. Firstly, BERT is fine-tuned on the specified task to get the teaching signal for distillation, and the student model with simpler architectures attempts to fit the task-specified fine-tuned BERT afterward. Such methodologies can achieve satisfying results by capturing the task-specified biases (McCallum and Nigam, 1999; Godbole et al., 2018; Min et al., 2019), which are inherited by the tuned BERT (Niven and Kao, 2019; McCoy et al., 2019). Unfortunately, the powerful generalization nature of BERT tends to be lost. Apparently, distilling BERT’s original motivation is to obtain a lightweight substitution of BERT for online implementations, and BERT’s general semantic knowledge, which plays a significant role in some NLP tasks like sentence similarity quantification, is expected to be maintained accordingly. Meanwhile, for many NLP tasks, manual labeling is quite a high-cost work, and large amounts of annotated data can not be guaranteed to obtain. Thus, it

\* Equal contribution during the internship at Tencent.

is of great necessity to compress BERT with the non-task-specific training procedure on unlabeled datasets.

For achieving the Non-task-specific Distillation from BERT, this paper proposes a distillation loss function to approximate sentence representations by minimizing the cosine distance between the sentence representation given by the student network and the one from BERT. As a result, a student network with a much smaller scale of parameters is produced. Since the distilling strategy purely focuses on the simulation of sentence embeddings from BERT, which is not directly related to any specific NLP task, the whole training procedure takes only a large amount of sentences without any manual labeling work. Similar to BERT, the smaller student network can also perform transfer learning to any sentence-level downstream tasks, such as text classification and sentence matching. The proposed methodology is evaluated on the open platform of General Language Understanding Evaluation (GLUE) (Wang et al., 2019), including the Single Sentence (SST-2), Similarity and Paraphrase (QQP and MRPC), and Natural Language Inference (MNLI) tasks. The experimental results show that our proposed model outperforms the models distilled from a BERT fine-tuned on a specific task. Moreover, our model inferences more efficiently than other transformer-based distilled models.

## 2 Related Works

With the propose of ELMo (Peters et al., 2018), various studies take the representation given by pre-trained language models as additional features to improve the performances. Howard and Ruder (2018) propose Universal Language Model Fine-tuning (ULMFiT), an effective transfer learning method that can be applied to any NLP task and accordingly, using pre-trained language models in downstream tasks became one of the most exciting directions. On this basis, developing with deeper network design and more effective training methods, pre-trained models’ performances improved continuously (Devlin et al., 2019; Radford et al., 2019; Yang et al., 2019; Liu et al., 2019b). Since the release of BERT (Devlin et al., 2019), the state-of-the-art (SOTA) results on 11 NLP tasks have been produced consequently.

With the improvement in performances, the computing cost increases, and the inference procedure becomes slower accordingly. Thus, various stud-

ies focused on the model compression upon BERT. Among the most common model compression techniques, the knowledge distillation (Hinton et al., 2015) has been proven to be efficient in transferring the knowledge from large-scaled pre-trained language models into another one (Liu et al., 2019a; Wang et al., 2020; Jiao et al., 2019; Sun et al., 2020). With the help of proposed distillation loss, Sun et al. (2019) compressed BERT into fewer layers by shortening the distance of internal representations between student and teacher BERTs. For the sentence-pair modeling, Barkan et al. (2019) found the cross-attention function across sentences is consuming and tried to remove it with distillation on sentence-pair tasks. Different from these studies distilling BERT into transformer-based models, Chia et al. (2019) proposed convolutional student architecture to distill GPT for efficient text classification. Moreover, focusing on the sequence labeling tasks, (Tsai et al., 2019) derived a BiLSTM or MiniBERT from BERT via standard distillation procedure to simulate the prediction on each token. Besides, Tang et al. (2019a,b) proposed to distill BERT into a BiLSTM based model with penalizing the mean square error between the student’s logits and the ones given by BERT as the objective on specific tasks, and introduced various data augmentation methods during distillation.

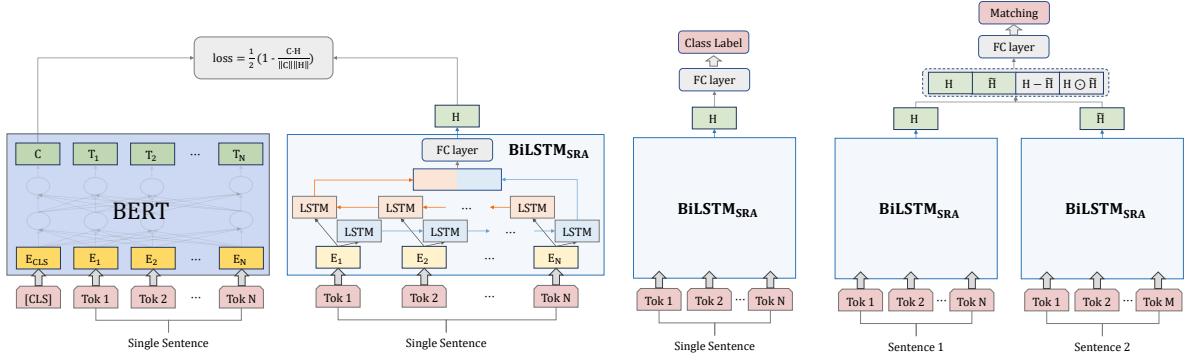
## 3 Method

As introduced in Section 1, our proposed method consists of two procedures. Firstly, we distill BERT into a smaller student model via approximating the representation of sentences given by BERT. Afterward, similar to BERT, the student model can be fine-tuned on any sentence-level task, such as text classification and sentence matching.

### 3.1 Distillation Procedure

Suppose  $x = \{w_1, w_2, \dots, w_i, \dots w_n | i \in [1, n]\}$  stands for a sentence containing  $n$  tokens ( $w_i$  is the  $i$ -th token of  $x$ ), and let  $T : x \rightarrow T_x \in \mathbb{R}^d$  be the teacher model which encodes  $x$  into  $d$ -dimensional sentence embedding  $T_x$ , the goal of the sentence approximation oriented distillation is to train a student model  $S : x \rightarrow S_x \in \mathbb{R}^d$  generating  $S_x$  as the approximation of  $T_x$ .

In our proposed distillation architecture, as shown in Figure 1a, we take the BERT as the teacher model  $T$ , and the hidden representation  $C$  is extracted from the top transformer layer upon



(a) Distilling BERT based on Representation Approximation. (b) Tuning (sentence classification). (c) Tuning (sentence-pair oriented).

Figure 1: The illustration of the proposed BERT distillation architecture including the distilling and tuning procedures. Sub-figure (a) demonstrates the distillation procedure taking BERT as the teacher model and BiLSTM as the student model, with the objective of approximating the sentence representations given by BERT. (b) and (c) show two types of fine-tuning frameworks, in which (b) addresses the sentence classification task with the single sentence as the input, and (c) goes for the sentence-pair-oriented tasks, i.e., sentence similarity quantification, natural language inference.

the [CLS]<sup>1</sup> token as  $T_x$ . For the student model, a standard bidirectional LSTM (BiLSTM) is first employed to encode the sentence into a fixed-size vector  $H$ . After that, a fully connected layer without bias terms is built upon the BiLSTM layer to map  $H$  into a  $d$ -dimensional representation, followed by a  $\tanh$  activation that normalizes the values of previous representation between -1 and 1 as the final  $S_x$ .

As our non-task-specific distillation task has no labeling data, and the signal given by the teacher is a real value vector, it is not feasible to minimize the cross-entropy loss over the soft labels and ground truth labels (Sun et al., 2019; Barkan et al., 2019; Tang et al., 2019b). On this basis, we propose an adjusted cosine similarity between the two real value vectors  $T_x$  and  $S_x$  to perform the sentence representation approximation. Our distillation objective is computed as follows:

$$\mathcal{L}_{distill} = \frac{1}{2}(1 - \frac{T_x \cdot S_x}{\|T_x\| \|S_x\|}) \quad (1)$$

Here  $\tanh$  is chosen as the activation function since most values (more than 98% according to our statistics) in  $T_x$  obtained from BERT are within range of  $\tanh$  (-1 to 1). The choice of using cosine similarity based loss is mainly based on the following two considerations. Firstly, since 2% values in  $T_x$  are outside the range of [-1, 1], it is more reasonable

<sup>1</sup>[CLS] is a special symbol added in front of other tokens in BERT, and the final hidden state corresponding to this token is usually used as the aggregate sequence representation.

to use a scalable measurement, such as cosine similarity, to deal with these deviations. Secondly, it is meaningful to compute the cosine similarity between sentence embeddings given by BERT (Xiao, 2018).

Overall, after the distillation procedure, we obtained a BiLSTM based “BERT”, which is smaller in parameter scale and more efficient in generating a sentence’s semantic representation.

**Distilling data** As our distillation procedure needs no dependency on sentence type or labeling resources but only standard sentences available everywhere, the distillation data selection follows the existing literature on language model pre-training as well as BERT. We use the English Wikipedia to perform the distillation. Furthermore, as the proposed method focus on the sentence representation approximation, the document is segmented into sentences using spacy (Honnibal and Montani, 2017).

### 3.2 Fine-tuning the Student Model

The fine-tuning on sentence-level tasks is straightforward. The downstream tasks discussed in this paper can be summarized as type judgment on a single sentence and predicting the relationship between two sentences (same as all GLUE tasks). Figure 1b illustrates the model architecture for single sentence classification tasks. The student model  $S$  is utilized to provide sentence representation. After that, a multilayer perceptron (MLP) based classifier

using Relu as activation of hidden layers is applied for the specific task. For the sentence pair tasks, as shown in Figure 1c, the representations  $H$  and  $\tilde{H}$  for the sentence pair are obtained by transforming two sentences into two BiLSTM based student models with shared weights respectively. Then, following the baseline BiLSTM model reported by GLUE (Wang et al., 2019), we apply a standard concatenate-compare operation between the two sentence embeddings and get an interactive vector as  $[H, \tilde{H}, |H - \tilde{H}|, H \odot \tilde{H}]$ , where the  $\odot$  demotes for the element-wise multiplication. Then, same as the single sentence task, an MLP based classifier is built upon the interactive representation.

For both types of tasks, MLP layers are initialized randomly, and the rest parameters are inherited from the distilled student model. Meanwhile, all parameters are optimized through the training procedure for the specific task.

## 4 Experimental Setups

### 4.1 Datasets & Evaluation Tasks

To evaluate the performance of our proposed non-task-specific distilling method, we conduct experiments on three types of sentence-level tasks: sentiment classification (SST-2), similarity (QQP, MRPC), and natural language inference (MNLI). All the tasks come from the GLUE benchmark (Wang et al., 2019).

**SST-2** Based on the Stanford Sentiment Treebank dataset (Socher et al., 2013), the SST-2 task is to predict the binary sentiment of a given single sentence. The dataset contains 64k sentences for training and remains 1k for testing.

**QQP** The Quora Question Pairs<sup>2</sup> dataset consists of pairs of questions, and the corresponding task is to determine whether each pair is semantically equivalent.

**MNLI** The Multi-Genre Language Inference Corpus (Williams et al., 2018) is a crowdsourced collection of sentence pairs with textual entailment annotations. There are two sections of the test dataset: matched (in-domain, noted as MNLI-m) and mismatched (cross-domain, noted as MNLI-mm).

**MRPC** The Microsoft Research Paraphrase Corpus (Dolan and Brockett, 2005) is similar to the

QQP dataset. This dataset consists of sentence pairs with binary labels denoting their semantic equivalence.

### 4.2 Model Variations

**BERT** (Devlin et al., 2019) with two variants: BERT<sub>BASE</sub> and BERT<sub>LARGE</sub>, containing 12 and 24 layers of Transformer respectively.

**ELMO Baseline** (Wang et al., 2019) is a BiLSTM based model, taking ELMo (Peters et al., 2018) embeddings in place of word embeddings.

**BERT-PKD** (Sun et al., 2019) proposes a patient knowledge distillation approach to compress BERT into a BERT with fewer layers. BERT<sub>3</sub>-PKD and BERT<sub>6</sub>-PKD stand for the student models consisting of 3 and 6 layers of Transformer, respectively.

**DSE** (Barkan et al., 2019) is a sentence embedding model based on knowledge distillation from cross-attentive models. For each single sentence modeling, the 24-layers BERT is employed.

**BiLSTM<sub>KD</sub>** (Tang et al., 2019b) introduces a new distillation objective to distill a BiLSTM based model from BERT for a specific task. BiLSTM<sub>KD</sub>+TS (Tang et al., 2019a) donates the distilling procedure performed with the proposed data augmentation strategies.

**BiLSTM<sub>SRA</sub>** stands for the Sentence Representation Approximation based distillation model proposed in this paper. BiLSTM<sub>SRA</sub> + KD donates performing knowledge distillation method proposed by Tang et al. (2019b) during fine-tuning on a specific task, and BiLSTM<sub>SRA</sub> + KD+TS demonstrates using the same augmented dataset to perform the distillation.

### 4.3 Hyperparameters

For the student model in our proposed distilling method, we employ the 300-dimension GloVe (840B Common Crawl version; Pennington et al., 2014) to initialize the word embeddings. The number of hidden units for the bi-directional LSTM is set to 512, and the size of the task-specific layers is set to 256. All the models are optimized using Adam (Kingma and Ba, 2015). In the distilling procedure, we choose the learning rate as  $1 \times 10^{-3}$  with the batch size=1024. During fine-tuning, the best learning rate on the validation set is picked from  $\{2, 3, 5, 10\} \times 10^{-4}$ . For the data

<sup>2</sup><https://www.quora.com/q/quoradata/First-Quora-Dataset-Release-Question-Pairs>

#	Models	SST-2	QQP	MNLI-m/mm	MRPC
		Acc	F <sub>1</sub> /Acc	Acc	F <sub>1</sub> /Acc
1	BiLSTM (report by GLUE)	85.9	61.4/81.7	70.3/70.8	79.4/69.3
2	BiLSTM (report by Tang et al. (2019b))	86.7	63.7/86.2	68.7/68.3	80.9/69.4
3	BiLSTM (our implementation)	84.5	60.3/81.6	70.8/69.4	80.2/69.7
4	ELMO Baseline (Wang et al., 2019)	90.2	65.6/85.7	72.9/73.4	84.9/78.0
5	BERT <sub>BASE</sub> (Devlin et al., 2019)	93.5	71.2/89.2	84.6/83.4	88.9/84.8
6	BERT <sub>LARGE</sub> (Devlin et al., 2019)	94.9	72.1/89.3	86.7/85.9	89.3/85.4
7	DSE (Barkan et al., 2019)	-	68.5/86.9	80.9/80.4	86.7/80.7
8	BERT <sub>6</sub> -PKD (Sun et al., 2019)	92.0	70.7/88.9	81.5/81.0	85.0/79.9
9	BERT <sub>3</sub> -PKD (Sun et al., 2019)	87.5	68.1/87.8	76.7/76.3	80.7/72.5
10	BiLSTM <sub>KD</sub> (Tang et al., 2019a)	88.4	-/-	-/-	78.0/69.7
11	BiLSTM <sub>SRA</sub> ( <b>Ours</b> )	90.0	64.4/86.2	<b>72.6/72.5</b>	<b>83.1/75.1</b>
12	BiLSTM <sub>SRA</sub> + KD	<b>90.2</b>	<b>67.7/87.8</b>	72.3/72.0	80.2/72.8
13	BiLSTM <sub>KD+TS</sub> (Tang et al., 2019b)	90.7	68.2/88.1	<b>73.0/72.6</b>	82.4/76.1
14	BiLSTM <sub>SRA</sub> + KD+TS	<b>91.1</b>	<b>68.4/88.6</b>	<b>73.0/72.9</b>	<b>83.8/76.2</b>
Improvements obtained by performing different knowledge distillations					
15	PKD (Sun et al., 2019)	+1.1	+2.3/+0.9	<b>+1.9/+2.0</b>	+0.2/-0.1
16	KD (Tang et al., 2019a)	+1.7	-/-	-/-	-2.9/+0.3
17	SRA( <b>Ours</b> )	+5.5	+4.1/+4.6	+1.8/+3.1	<b>+2.9/+5.4</b>
18	SRA( <b>Ours</b> )+KD	<b>+5.7</b>	<b>+7.4/+6.2</b>	+1.5/+2.6	0./+3.1
19	KD+TS (Tang et al., 2019a)	+4.0	+4.5/+1.9	<b>+4.3/+4.2</b>	+1.5/+6.7
20	SRA( <b>Ours</b> )+KD+TS	<b>+6.6</b>	<b>+8.1/+7.0</b>	+2.2/+3.5	<b>+3.6/+6.5</b>

Table 1: Evaluation results with scores given by the official evaluation server<sup>3</sup>.

augmentation, we use the rule-based method originally suggested by Tang et al. (2019b). Notably, on the SST-2 and MRPC dataset, we stop data augmenting when the transfer set achieves 800K samples following the setting of their follow-up research (Tang et al., 2019a). Besides, inspired by the comparisons in the research of Sun et al. (2019), we find BERT<sub>BASE</sub> can provide more instructive representations than BERT<sub>LARGE</sub>. So that, we chose BERT<sub>BASE</sub> as our teacher model to train the non-task-specified BiLSTM<sub>SRA</sub>.

## 5 Results and Analysis

### 5.1 Model Performance Analysis

For a comprehensive experiment analysis, we collect data and implement comparative experiments on various published BERT and BERT-distillation methods. Table 1 shows the results of our proposed BiLSTM<sub>SRA</sub> and the baselines on the four datasets. All models in the first block (row 1-6) belong to base methods without implementing distillation, the second (row 7-9) and third (row 10-12) blocks

show the performances of distillation models using BERT and BiLSTM structures, respectively. Moreover, the fourth block (row 13-14) displays the influences of textual data augmentation approach on our BiLSTM<sub>SRA</sub> and BiLSTM<sub>KD</sub> distillation baseline. The last two blocks contain the results of pure improvements obtained by different distillation methods. To analyze the effectiveness of BiLSTM<sub>SRA</sub> thoroughly, we break down the analyses into the following two perspectives.

#### 5.1.1 Comparison Between Models

Taking those non-distillation methods in the first block as references, BiLSTM<sub>SRA</sub> performs on par with ELMO on all tasks. Especially, BiLSTM<sub>SRA</sub> + KD+TS outperforms the ELMO baseline by approximately 3% on QQP and 1% on SST-2 (row 14 vs 4). Such fact shows our compressed “BERT” can provide as good pre-trained representations as ELMO on the sentence-level tasks.

For those distillation methods, both our model and BiLSTM<sub>KD</sub> distill knowledge from BERT into a simple BiLSTM based model, while BERT-PKD focuses on distilling with the BERT of fewer lay-

<sup>3</sup><https://gluebenchmark.com/leaderboard>

ers. Despite the powerful BERT based student model and large-scale parameters used by BERT-PKD, our proposed BiLSTM<sub>SRA</sub> still outperforms BERT<sub>3</sub>-PKD on SST-2 and MRPC dataset (row 12 vs. 9). For BiLSTM<sub>KD</sub>, it proposes a rule-based textual data augmentation approach (noted as TS) to construct transfer sets for the task-specific knowledge distillation. We also employ such method upon BiLSTM<sub>SRA+KD</sub>. With and without the data augmentation, BiLSTM<sub>SRA</sub> consistently outperforms BiLSTM<sub>KD</sub> on all tasks (row 12 vs 10; row 14 vs 13). Coworking with the standard knowledge distillation and data augmentation methods, our proposed model is sufficient to distill semantic representation modeled from pre-training tasks as well as the task-specific knowledge included in a fine-tuned BERT.

Besides, DSE’s overall architecture is similar to our method for modeling the sentence matching task, except DSE does not reduce the parameter size because it employs the pre-trained BERT<sub>LARGE</sub> to give sentence representations. Thus, on the sentence-pair level tasks, DSE somehow is an upper bound of the distilled models without utilizing any cross attention to model the two sentences’ interaction. Comparing with DSE achieved an averaged 80.7 score on all sentence-pair level tasks, BiLSTM<sub>SRA+KD+TS</sub> can also obtain 77.2 that only 3.5 points lower (row 7 vs. 14). Analyzing from this fact, our proposed model has distilled a much smaller “BERT” with acceptable performances.

### 5.1.2 Distillation Effectiveness

Because in each paper, the performances of student models used for distillation vary from each other. To further evaluate the distillation effectiveness, we also report each distillation method’s improvement upon the corresponding student directly trained without distillation (in row 15-20). It can be observed that SRA improves the scores by over 3.9% on average, while PKD and KD only provide less than 1.2% increase (row 17/16 vs. 15).

Since our distillation method is unrelated to specific tasks, KD can also be performed upon BiLSTM<sub>SRA</sub> during fine-tuning on a given dataset. This operation provides a notable boost on the QQP task, but damages the performance on both MNLI and MRPC datasets (row 17 vs. 18). We attribute these differences to the following aspects: a) the QQP dataset has more obvious task-specified bi-

Models	# of Par.	Inference Time
BERT <sub>LARGE</sub>	309 (64x)	1461.9 (54.4x)
BERT <sub>BASE</sub>	87 (18x)	479.7 (17.7x)
ELMO	93 (19x)	-(23.7x)
BERT <sub>3</sub> -PKD	21 (4x)	-(4.8x)
BERT <sub>6</sub> -PKD	42 (9x)	-(9.2x)
DSE	309 (64x)	-(109.1x)
BiLSTM <sub>KD</sub>	<b>2.4 (0.5x)</b>	31.9 (1.2x)
BiLSTM <sub>SRA</sub>	4.8 (1x)	<b>26.8 (1x)</b>

Table 2: Comparisons of model size and inference speed. # of Par. denotes the number of millions of parameters, and the inference time is in seconds. The factors inside the brackets are computed comparing to our proposed model.

ases during the sampling process<sup>4</sup>. A pre-trained BERT can not learn such biases; b) a fine-tuned BERT on the MNLI can not further provide more easy-to-use information to guide the student training after performing SRA; c) MRPC does not include enough data to complete KD, which is also indicated by the decreased F1 score shown in row 16 in Table 1. These phenomena reflect that the pre-distillation without paying attention to a specific task can help to learn more useful semantic information from the teacher model.

Different from obtaining the best results on the MNLI dataset, SRA+KD+TS brings few improvements compared to KS+TS (row 19 vs. 20). We attribute this to the difference in the results of pure student BiLSTM between our implementation and the one of Tang et al. (2019b), though our scores are more constant with the baselines given by the GLUE benchmark (Wang et al., 2019).

### 5.2 Model Efficiency Analysis

To compare the inference speeds of different models, we also implement experiments on 100k samples from the QQP dataset. The results are shown in Table 2. All the inference procedures are performed on a single P40 GPU with a batch size of 1024, respectively. As the inference time is affected by the test machine’s computing power, for fair comparisons with ELMO, BERT<sub>3</sub>-PKD, BERT<sub>6</sub>-PKD, and DSE, we inherit the speed-up factors from previous papers. Besides, the numbers of parameters reported in Table 2 exclude those

<sup>4</sup><https://www.kaggle.com/c/quora-question-pairs/discussion/32819#latest-189493>

Models	20%	30%	50%	100%
BERT <sub>LARGE</sub>	91.9	92.5	93.5	93.7
BiLSTM	80.7	81.0	83.6	84.5
BiLSTM <sub>KD</sub>	81.9	83.2	84.8	86.3
BiLSTM <sub>SRA</sub>	<b>85.9</b>	<b>87.3</b>	<b>88.1</b>	<b>89.2</b>

Table 3: The accuracy scores evaluated on the SST-2 validation set. The models are trained with different proportions of the training data.

from the embedding layers, since such components do not affect the inference speed and are positively related to the vocabulary sizes, i.e., usually few words appeared for a specific task.

From the results shown in Table 2, it can be observed that the BiLSTM based distilled models have fewer parameters than BERT, ELMO, as well as the other transformer-based models. Compared to the lightest model, both the BERT<sub>BASE</sub> and ELMO are around 20 times larger in parameter size and 20 times slower in inference speed. Even the smallest transformer based model BERT<sub>3-PKD</sub> is also four times larger than our proposed BiLSTM<sub>SRA</sub>. Comparing with BiLSTM<sub>KD</sub>, although our proposed BiLSTM<sub>SRA</sub> is larger in parameter size due to the restriction of the sentence embedding’s dimension given by the teacher BERT, it stills inferences more efficiently. This is mainly due to the fact that the more hidden units in BiLSTM<sub>SRA</sub> are more accessible to calculated in parallel by the GPU core, while the larger word embedding size in BiLSTM<sub>KD</sub> slows down its inference efficiency. In conclusion, the cost and production per second of BiLSTM<sub>KD</sub> and BiLSTM<sub>SRA</sub> are within the same scale, but our method achieves better results on GLUE tasks according to the comparison shown in Table 1.

### 5.3 Influence of Task-specific Data Size

Since pre-trained language models have well-initialized parameters and only learn a few parameters from scratch, these models usually converge faster and are less dependent on large-scale annotations. Correspondingly, the non-task-specific distillation method proposed in this paper also aims to obtain a compressed pre-trained BERT and keep these desirable properties. To evaluate it, in this section, we discuss the influence of the task-specific training data and learning iterations on the performance of our model and the others.

As illustrated in Table 3, we experiment in train-

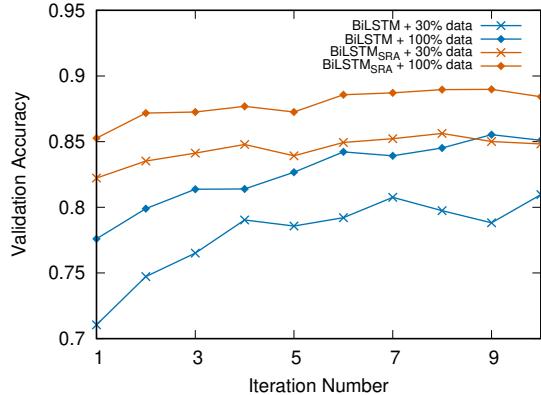


Figure 2: Learning curve on the QQP dataset.

ing the models using different proportions of the dataset. BERT<sub>LARGE</sub> trained on the corresponding data stands for the teacher model of each BiLSTM<sub>KD</sub>. No doubt, all the models can achieve better results using more training data, while BERT performs the best. BERT even successfully predicts 91.9% of validation samples under only 20% training data. Comparing with the pure BiLSTM models, the BiLSTM<sub>KD</sub> models slightly improve the performances by 1%~2%, whereas BiLSTM<sub>SRA</sub> outperforms the best BiLSTM model as well as the BiLSTM<sub>KD</sub> trained with 20% and 30% percent data respectively. Besides, similar to BERT, the difference of accuracy between BiLSTM<sub>SRA</sub> trained with 20% and the one using 100% corpus is relatively small. This phenomenon indicates that our model converges faster and is less dependent on the amount of training data for downstream tasks.

Such conclusions are also reflected in the comparison in Figure 2 of the models’ learning curves on QQP. Even though QQP is a large dataset to train a good BiLSTM model, it can be observed that BiLSTM<sub>SRA</sub> trained with 30% data performs equivalent to BiLSTM using the whole corpus. Moreover, using 100% training data, BiLSTM<sub>SRA</sub> even outperforms the converged BiLSTM after the first epoch. Besides, all the BiLSTM<sub>SRA</sub> models converge in much fewer epochs.

### 5.4 Influence of Distilling Data Size

Despite the task-specified data, Wikipedia corpus is used in the distillation procedure of our proposed method. We also pre-train different BiLSTM<sub>SRA</sub> base models using {1, 2, 4} million Wikipedia data, and the corresponding fine-tuning performances on SST-2 and MNLI are reported in Table 4. It can be observed that both the performances of

Size	Distillation	SST-2	MNLI-m
	Loss	Acc	Acc
0M	-	84.5	70.23
1M	0.0288	88.9 (+4.4)	72.01 (+1.78)
2M	0.0257	89.3 (+4.8)	72.09 (+1.86)
4M	<b>0.0241</b>	<b>89.4 (+4.9)</b>	<b>72.45 (+2.22)</b>

Table 4: The distillation losses on the Wikipedia validation set and the accuracy scores of the downstream tasks various with the distillation data sizes.

BiLSTM<sub>SRA</sub> on SST-2 and MNLI are proportional to the distillation loss. This observation indicates the effectiveness of our proposed distillation process and objective.

Besides, distilling with adequate data is sufficient to produce more BERT-like sentence representations as well as achieve better performance on the downstream tasks. Nevertheless, different from the fact that more training data has a significant benefit in a particular task, four times the distilling data can only improve around 0.5 points on both SST-2 and MNLI-m tasks. Thus, our method does not require a vast amount of training data and a long training time to obtain good sentence representations. Furthermore, the second column’s loss scores suggest BiLSTM<sub>SRA</sub> can generate more than 95% similar sentence embeddings with the ones given by BERT under the measure of the cosine similarity.

### 5.5 Analysis on the Untuned Sentence Representations

A notable characteristic of the pre-trained language models, such as ELMO, BERT, and certainly the non-task oriented distillation models, lies in the capability of providing sentence representations for quantifying similarities of sentences, without any tuning operation based on specific tasks. In this subsection, we conduct the comparisons among models by directly extracting their sentence embeddings without fine-tuning upon sentence similarity oriented tasks.

Table 5 lists the results of models on the QQP dataset. It should be noted that, in this table, ELMO, BERT<sub>BASE</sub> (CLS) and BERT<sub>BASE</sub> (averaged) are introduced as the comparison basis, since they can give the SOTA untuned sentence representations for the similarity measurement. The comparison mainly focuses on the performances of

Models	Acc	F <sub>1</sub>
ELMO	65.1	<b>64.4</b>
BERT <sub>BASE</sub> (CLS)	63.9	61.0
BERT <sub>BASE</sub> (averaged)	<b>66.4</b>	64.1
BiLSTM <sub>KD</sub>	56.3	56.6
BiLSTM <sub>SRA</sub>	<b>62.9</b>	<b>61.0</b>

Table 5: Results of untuned sentence representing models on QQP dataset.

our proposed BiLSTM<sub>SRA</sub> and BiLSTM<sub>KD</sub>. For a thorough comparison, we define the training objective of BiLSTM<sub>KD</sub> as fitting the cosine similarity score of the sentence pair directly given by the pre-trained BERT<sub>BASE</sub>, which means both the teacher BERT and distilled models do not utilize the labels of QQP dataset. Even though the training goal of BiLSTM<sub>KD</sub> is more direct than BiLSTM<sub>SRA</sub>, it can be seen that our BiLSTM<sub>SRA</sub> outperforms the former on the metrics. Furthermore, it achieves scores closed to those of BERT<sub>BASE</sub>. Besides, we can also observe that, for sentence similarity quantification, averaging the context word embeddings as the sentence representation (ELMO and BERT<sub>BASE</sub> (averaged)) works better than taking the final hidden state corresponding to the [CLS] token (BERT<sub>BASE</sub> (CLS)).

## 6 Conclusions

In this paper, we have presented a sentence representation approximating oriented method for distilling the pre-trained BERT model into a much smaller BiLSTM without specifying tasks, so as to inherit the general semantic knowledge of BERT for better generalization and universal-usability. The experiments conducted based on the GLUE benchmark have shown that our proposed non-task-specific distillation methodology can improve the performances on multiple sentence-level downstream tasks. From the experimental results, the following conclusions can be drawn: 1) for a specified task, our proposed distillation method can bring the 5% improvement to the pure BiLSTM model on average; 2) the proposed model can outperform the state-of-the-art BiLSTM based pre-trained language model, which contains much more parameters; 3) compared to the task-specific distillation, our distilled model is less dependent on the corpus size of the downstream task with satisfying performances guaranteed.

## References

- Oren Barkan, Noam Razin, Itzik Malkiel, Ori Katz, Avi Caciularu, and Noam Koenigstein. 2019. Scalable attentive sentence-pair modeling via distilled sentence embedding. *arXiv preprint arXiv:1908.05161*.
- Yew Ken Chia, Sam Witteveen, and Martin Andrews. 2019. Transformer to cnn: Label-scarce distillation for efficient text classification. *arXiv preprint arXiv:1909.03508*.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.
- William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.
- Ameya Godbole, Aman Dalmia, and Sunil Kumar Sahu. 2018. Siamese neural networks with random forest for detecting duplicate question pairs. *arXiv preprint arXiv:1801.07288*.
- Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.
- Matthew Honnibal and Ines Montani. 2017. spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing. *To appear*, 7.
- Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339.
- Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. **Tinybert: Distilling BERT for natural language understanding.** *CoRR*, abs/1909.10351.
- Diederik P. Kingma and Jimmy Ba. 2015. **Adam: A method for stochastic optimization.** In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.
- Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019a. Improving multi-task deep neural networks via knowledge distillation for natural language understanding. *arXiv preprint arXiv:1904.09482*.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.
- Andrew McCallum and Kamal Nigam. 1999. Text classification by bootstrapping with keywords, em and shrinkage. In *Unsupervised Learning in Natural Language Processing*.
- Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. **Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference.** In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 3428–3448.
- Sewon Min, Eric Wallace, Sameer Singh, Matt Gardner, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019. Compositional questions do not necessitate multi-hop reasoning. *arXiv preprint arXiv:1906.02900*.
- Timothy Niven and Hung-Yu Kao. 2019. Probing neural network comprehension of natural language arguments. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4658–4664.
- Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.
- Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of NAACL-HLT*, pages 2227–2237.
- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8).
- Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.
- Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for bert model compression. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4314–4323.
- Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. **Mobilebert: a compact task-agnostic BERT for resource-limited devices.** In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*,

*ACL 2020, Online, July 5-10, 2020*, pages 2158–2170. Association for Computational Linguistics.

Raphael Tang, Yao Lu, and Lin Jimmy. 2019a. Natural language generation for effective knowledge distillation. In *Proceedings of the Workshop on Deep Learning Approaches for Low-Resource NLP*.

Raphael Tang, Yao Lu, Linqing Liu, Lili Mou, Olga Vechtomova, and Jimmy Lin. 2019b. Distilling task-specific knowledge from bert into simple neural networks. *arXiv preprint arXiv:1903.12136*.

Henry Tsai, Jason Riesa, Melvin Johnson, Naveen Ariavazhagan, Xin Li, and Amelia Archer. 2019. Small and practical bert models for sequence labeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3623–3627.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *arXiv: Computation and Language*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122.

Han Xiao. 2018. bert-as-service. <https://github.com/hanxiao/bert-as-service>.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*.

# A Simple and Effective Usage of Word Clusters for CBOW Model

**Yukun Feng<sup>1</sup>, Chenlong Hu<sup>1</sup>, Hidetaka Kamigaito<sup>1</sup>, Hiroya Takamura<sup>1,2</sup> and Manabu Okumura<sup>1</sup>**

<sup>1</sup>Tokyo Institute of Technology

<sup>2</sup>National Institute of Advanced Industrial Science and Technology (AIST)

{yukun, huchenlong, kamigaito, oku}@lr.pi.titech.ac.jp  
takamura@pi.titech.ac.jp

## Abstract

We propose a simple and effective method for incorporating word clusters into the Continuous Bag-of-Words (CBOW) model. Specifically, we propose to replace infrequent input and output words in CBOW model with their clusters. The resulting cluster-incorporated CBOW model produces embeddings of frequent words and a small amount of cluster embeddings, which will be fine-tuned in downstream tasks. We empirically show our replacing method works well on several downstream tasks. Through our analysis, we show that our method might be also useful for other similar models which produce word embeddings.

## 1 Introduction

Word embeddings have been widely applied to various natural language processing (NLP) tasks. These embeddings can be pretrained on a large corpus and carry useful semantic information. One of the most well-known methods for obtaining word embeddings is based on Continuous Bag-of-Words (CBOW) (Mikolov et al., 2013a) and there have been many research efforts to extend it.

In this paper, we focus on incorporating word clusters into CBOW model. Each word cluster consists of words that function similarly. By aggregating such words, we can alleviate data sparsity, even though each of those words is infrequent. In the past few years, word clusters have been applied to various tasks, such as named-entity recognition (Ritter et al., 2011), machine translation (Wuebker et al., 2013) and parsing (Kong et al., 2014). Many word clustering algorithms can be applied to a raw corpus with different languages and help us obtain word clusters easily without additional language resources.

In our method, we keep only very frequent words and replace the other words with their clusters for both input and output words in the CBOW model.

This is motivated by the fact that word clusters are more reliable than infrequent words. Thus, only very frequent word embeddings and a small amount of cluster embeddings are produced as the output. When fine-tuning the trained embeddings on downstream tasks, the embeddings of infrequent words within one cluster are initialized by the embedding of their cluster to increase the coverage of pretrained word embeddings.

Since word embeddings are usually trained on the large-scale dataset. For making clusters on the large-scale dataset, we choose bidirectional, interpolated, refining, and alternating (BIRA) predictive exchange algorithm (Dehdari et al., 2016)<sup>1</sup> as our clustering method. Because BIRA was reported to be faster than many other methods. Notably, it can produce 800 clusters on 1 billion English tokens in 1.4 hours.

We evaluate our cluster-incorporated word embeddings<sup>2</sup> on downstream tasks, in which *fine-tuning* of word embeddings is involved. The evaluation for frequent words, for which our method also works well, on word similarity tasks can be found in appendix A. For the downstream tasks, we choose language modeling (LM) tasks, which are a fundamental task in NLP, as well as two machine translation (MT) tasks. To verify the effect of word clusters across different languages, 8 typologically diverse languages are further selected for the LM task. Finally, an analysis is provided for our method. In summary, our replacing method can be used to improve the embeddings of frequent and infrequent words, to reduce the number of word embeddings and to make training more effective.

<sup>1</sup>We used ClusterCat (<https://github.com/jonsafari/clustercat>) as the implementation.

<sup>2</sup><https://github.com/yukunfeng/cluster-cbow>

## 2 Related Work

A number of related research efforts have been done to help to learn better word embeddings aiming at different aspects. For example, Neelakantan et al. (2014) proposed an extension that learns multiple embeddings per word type. Ammar et al. (2016) proposed methods for estimating embeddings for different languages in a single shared embedding space. There is also a lot of work that incorporates internal information of words, such as character-level information (Chen et al., 2015; Bojanowski et al., 2017) and morpheme information (Luong et al., 2013; Qiu et al., 2014). Our research aims at another aspect and focuses on incorporating word clusters into the CBOW model, which has not been studied before.

There have also been some previous researches that utilized word clusters for reducing the number of word embeddings. Botha et al. (2017) used word clusters to reduce the network size for the part-of-speech tagging task. Shu and Nakayama (2018) attempted to compress word embeddings without losing performance by constructing the embeddings with a few basic vectors. Our goal is different from the previous work in that we attempt to learn better word embeddings and do not aim at reducing the parameters when our embeddings are fine-tuned in downstream tasks. Nonetheless, the reduction of the number of word embeddings from the CBOW model before fine-tuning is still one of our goals as we can save space to store these embeddings and save time to download them. For example, Google News Vectors have around 3 million words, and we need only 2% of the number of the word embeddings if we choose 100K most frequent words and 10K word clusters in our method.

## 3 Our Method

### 3.1 CBOW Model

Let  $w_t$  denote the  $t$ -th word in a given text. We adopt the basic CBOW model architecture for learning word embeddings. The CBOW model predicts the output word  $w_t$  given the input words in the window which precede or follow the output word. When the window size is 2, as an example, the input words are  $w_{t-2}, w_{t-1}, w_{t+1}, w_{t+2}$ . We denote the input and output embeddings of word  $w_i$  respectively as  $\vec{x}_i$  and  $\vec{o}_i$ . The CBOW model computes

the hidden representation as follows:

$$\vec{h} = \frac{1}{2c} \sum_{i=-c, i \neq 0}^c \vec{x}_{t+i}, \quad (1)$$

where  $c$  is the window size. We use negative sampling (Mikolov et al., 2013b) to train the CBOW model by maximizing the following objective function:

$$\log \sigma(\vec{h}^T \vec{o}_t) + \sum_{j=1}^k \log \sigma(-\vec{h}^T \vec{o}^j), \quad (2)$$

where  $k$  is the size of the negative sample,  $\vec{o}^j$  is the  $j$ -th noise word embedding and  $\sigma$  is the sigmoid function. Each word in the negative sample is drawn from the unigram distribution.

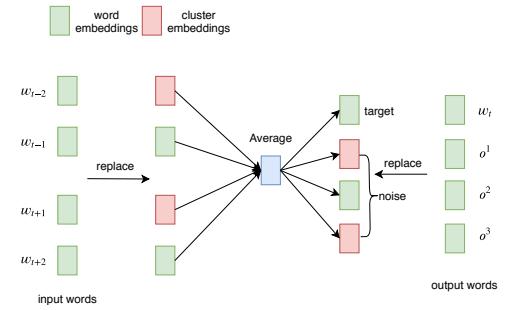


Figure 1: CBOW architecture with our replacing method for input and output words trained with negative sampling. Suppose that  $w_{t-2}, w_{t+1}, o^1$  and  $o^3$  are infrequent words.

### 3.2 Replacing Methods

As a method for incorporating word clusters, we propose to replace infrequent words with their clusters for the input and output. The architecture is shown in Figure 1. This is motivated by the intuition that the embeddings of clusters should be more reliable than those of infrequent words. We denote the embedding of the cluster for word  $w_{t+i}$  as  $\vec{d}_{t+i}$ . We present the following two replacing methods:

- **ReIn:** In the input,  $\vec{x}_{t+i}$  in Eq. (1) will be replaced with  $\vec{d}_{t+i}$  if the frequency of  $w_{t+i}$  is less than threshold  $f_{in}$ .
- **ReOut:** In the output, output words whose frequency is less than  $f_{out}$  are replaced with their clusters. Thus, in negative sampling, a noise word will be sampled from clusters and frequent words.

As with the standard CBOW model, we use the input word embeddings and input cluster embeddings for downstream tasks. Thresholds  $f_{in}$  and  $f_{out}$  are set to 100 in all experiments. Due to this large value, each cluster contains many infrequent words, which share the same embedding. We use two methods together, which is referred to as **ReIn+ReOut** in the following experiments.

### 3.2.1 Motivation of ReIn and ReOut

Since the embeddings of clusters are learned by aggregating many infrequent words, they are more robust than the embeddings of the infrequent words. During the fine-tuning process for a downstream task, the embeddings of infrequent words are first initialized with the embeddings of their clusters. As most of these infrequent words appear only a few times, these embeddings will not be updated far away from each other within one cluster. The visualization of these embeddings before and after fine-tuning can be found in the appendix B. As a result, these embeddings for infrequent words become more reliable since originally most infrequent word embeddings are updated only several times and are not far away from where they were randomly initialized. Since the context of frequent words becomes less noisy by replacing all the infrequent words with their clusters, the learned frequent word embeddings are also better, as shown later in our experiments.

The standard CBOW model is usually trained with negative sampling, which is designed for speeding up the training process. By using ReOut, infrequent noise words will be replaced with their clusters, which contain more noise words than the original CBOW model. As a result, ReOut makes the training of the CBOW model more effective, as shown later in our experiments.

## 4 Experiments on LM and MT

We applied our embeddings to downstream tasks: language modeling (LM) and low-resource machine translation (MT). When applying to the downstream tasks, we only used the training data of the specific task to obtain word clusters and embeddings without any extra data. We then used the learned embeddings to initialize the lookup table of word embeddings for the task. In this paper, we limit the applications of our model to relatively small datasets to demonstrate the usefulness of our method. We plan to conduct larger-scale experiments on more downstream tasks in future work. In

the following tables, CBOW and ReIn+ReOut indicate that they are initialization methods for specific downstream tasks.

### 4.1 Hyper-parameter Settings

In this section, we describe the hyper-parameters for producing word clusters and word embeddings. As we mentioned before, we obtained word clusters through the ClusterCat software. For most hyper-parameters, we used its default values. We set the number of clusters to 600 in all our experiments. Since our work involves many tasks in total, it is hard to choose the optimal number of word clusters for each task. We experimented with several values (600, 800 and 1000) and observed the same trend. Thus, we simply chose 600, for convenience, for all tasks. For producing word embeddings, our implementation was based on the fasttext<sup>3</sup>. Our cluster-incorporated CBOW model and the standard CBOW model were trained under the same hyper-parameters. We set most hyper-parameters as its default values. Namely, we set the training epoch to 5, the number of negative examples to 5, the window size to 5, and the minimum count of word occurrence to 5<sup>4</sup>.

### 4.2 LM on Standard English Datasets

We test ReIn+ReOut based on the recent state-of-the-art awd-lstm-lm codebase<sup>5</sup>(Merity et al., 2018) using two standard language modeling datasets: Penn Treebank (PTB) and WikiText-2 (Wiki2). We followed exactly the same setting in the source code. The results are shown in Table 1, and we found that our ReIn+ReOut is effective even with the strong baseline.

	PTB	Wiki2
AWD-LSTM w/o fine-tuning (Merity et al., 2018)	58.80	66.00
CBOW	58.39	65.48
ReIn+ReOut	<b>57.85</b>	<b>63.93</b>

Table 1: Perplexity results on PTB and Wiki2.

### 4.3 Low-resource NMT

We applied our method to the standard long-short term memory networks (LSTMs) based sequence-to-sequence (seq2seq) model on two datasets: German-English (de-en) with 153K sentence pairs

<sup>3</sup><https://github.com/facebookresearch/fastText>

<sup>4</sup>When we set the minimum count of word occurrence to 1, the standard CBOW does not perform well.

<sup>5</sup><https://github.com/salesforce/awd-lstm-lm>

from IWSLT 2014 (Cettolo et al., 2014), English-Vietnamese (en-vi) with 133K sentence pairs from IWSLT 2015 (Cettolo et al., 2012). The detailed data statistics of two low-resource NMT datasets is in Table 2. We used the opennmt-py toolkit<sup>6</sup> with a 2-layer bidirectional LSTM with hidden size of 500 and set the training epoch to 30. The word embedding size is set to 500 and the batch size is 64. We trained the seq2seq models by the SGD optimizer with start learning rate being 1.0, which will be decayed by 0.5 if perplexity does not decrease on the validation set. Other hyper-parameters were kept default. We also include some published results based on LSTM-based seq2seq models to gauge the result of our baseline. As shown in Table 3, without any extra language pair resources, the ReIn+ReOut initialization improves the BLEU score over the baseline by 1.29 and 0.51 points on de-en, en-vi respectively.

	de-en	en-vi
#Training pairs	153,348	133,317
#Test pairs	6,750	1,268
#Valid pairs	6,970	1,553
Train Vocab (source)	103,796	54,169
Train Vocab (target)	50,045	25,615

Table 2: Data statistics of two low-resource NMT datasets.

	de-en	en-vi
seq2seq with attention (Luong and Manning, 2015)	-	23.3
AC+LL (Bahdanau et al., 2017)	28.53	-
NPMT (Huang et al., 2018)	29.92	27.69
Our seq2seq with attention	28.95	28.16
CBOW	29.25	28.24
Our ReIn+ReOut	<b>30.24</b>	<b>28.67</b>

Table 3: BLEU scores on two low-resource MT datasets. NPMT in Huang et al. (2018) used a neural phrase-based machine translation model and AC+LL in Bahdanau et al. (2017) used a one-layer GRU encoder and decoder with attention.

#### 4.4 LM in Diverse Languages

To verify the effect of word clusters on different languages, we selected 8 datasets containing typologically diverse languages from LM datasets released by Gerz et al. (2018). The data statistics of 8 LM datasets is in Table 5. We basically used standard LSTMs instead of AWD-LSTM-LM to save time. We chose the available standard LSTM-LM code<sup>7</sup>. Hyper-parameters of our standard LSTM model on

<sup>6</sup><https://github.com/OpenNMT/OpenNMT-py>

<sup>7</sup>[https://github.com/pytorch/examples/tree/master/word\\_language\\_model](https://github.com/pytorch/examples/tree/master/word_language_model)

language modeling tasks is in Table 4. The results are shown in Table 6. Our LSTM-LM obtained better results than the one from Gerz et al. (2018) on all datasets. As we see, ReIn+ReOut is effective for typologically diverse languages and also requires a smaller input vocabulary. For example, the input vocabulary of ReIn+ReOut for en dataset contains 1.3K words while the full vocabulary 50K.

Embedding size	200
Epochs	40
LSTM layers	2
Optimizer	SGD
LSTM sequence length	35
Learning rate	20
LSTM hidden unit	200
Learning rate decay	4
Param. init: rand uniform	[-0.1,0.1]
Gradient clipping	0.25
Dropout	0.2
Batch size	20

Table 4: Hyper-parameters of our standard LSTM model on language modeling task.

## 5 Analysis

In this section, we analyse ReIn+ReOut on the basis of LM experiments with en and de datasets.

### 5.1 Targeted Perplexity Results

To show the gain for frequent and infrequent words, we measured the perplexity for frequent and infrequent words in the test data separately. Specifically, we calculated the perplexity of the next word, when an infrequent word is given as the current word. A similar analysis on language models can be found in Vania and Lopez (2017). Our analysis do not contain new words in the test dataset. The results are shown in Table 7. As we see, ReIn+ReOut is more effective than CBOW in learning both the embeddings of frequent and infrequent words, as we explained in Sec. 3.2.1.

### 5.2 Ablation Study

The results of ablation study are in Table 8. Comparing the methods ReIn and CBOW, we found replacing only input infrequent words in CBOW also works better than the original CBOW. We can also conclude that replacing only output infrequent words in CBOW works better than the original CBOW, by comparing ReOut and CBOW. Both ReIn and ReOut work well even when they are used alone. As mentioned in the motivation of ReOut, it makes the training more effective. To verify

	Typology	Train vocab	#Train tokens	#Test tokens	#Valid tokens	#Input vocab of ReIn+ReOut
zh (Chinese)	Isolating	43674	746K	56.8K	56.9K	1661
vi (Vietnamese)	Isolating	32065	754K	61.9K	64.8K	1716
de (German)	Fusional	80743	682K	51.3K	52.6K	1163
en (English)	Fusiona	55522	783K	59.5K	57.3K	1381
ar (Arabic)	Introflexive	89091	723K	54.7K	55.2K	1431
he (Hebrew)	Introflexive	83223	719K	54.7K	52.9K	1345
et (Estonian)	Agglutinative	94184	556K	38.6K	40.0K	1285
tr (Turkish)	Agglutinative	90847	627K	45.2K	47.4K	1241

Table 5: Data statistics of 8 language modeling datasets and size of input vocabulary of our ReIn+ReOut.

Dataset	Random	CBOW	ReIn+ReOut
zh	555	527	<b>494</b>
vi	153	145	<b>138</b>
de	609	542	<b>484</b>
en	365	317	<b>289</b>
ar	1647	1447	<b>1305</b>
he	1482	1236	<b>1175</b>
et	1451	1157	<b>1004</b>
tr	1379	1220	<b>1148</b>

Table 6: Perplexity results of standard LSTM LM on 8 datasets with different initialization methods.

		Freq.	Infreq.	All
en	CBOW	340	198	283
	ReIn+ReOut	316	184	264
de	CBOW	591	352	489
	ReIn+ReOut	564	318	458

Table 7: Targeted perplexity results of standard LSTM LM with different initializations.

this, we increased the number of negative samples for ReIn and CBOW. The training will be more effective if we increase the number of negative samples, while training the model will also take longer time. As we increased the size of negative samples, we obtained better results for both ReIn and CBOW. We increased it only to 30 because we did not observe improvements when we made it further larger. This result indicates that we can use word clusters to obtain better results with a small amount of negative samples. In reality, we can also use off-the-shelf word clusters to avoid spending time for producing word clusters.

	en	de		en	de
ReIn	300	528	CBOW	317	542
ReIn neg+10	293	499	CBOW neg+10	309	523
ReIn neg+30	300	494	CBOW neg+30	312	554
ReIn+ReOut	289	484	ReOut	312	515

Table 8: Perplexity results of LSTM LM by changing the number of negative samples. '+neg' represents the number of negative samples, which is 5 at default.

### 5.3 LM Results on Off-the-shelf Vectors

To gauge the improvements, we used off-the-shelf pretrained word vectors in English: GloVe vectors ([Pennington et al., 2014](#)) and Google News Vectors<sup>8</sup>. We obtained 258, 290 and 289 perplexity scores on en with Google News Vectors, Glove vectors and ReIn+ReOut respectively. Although ReIn+ReOut underperforms Google News Vectors, which were trained on 100 billion tokens, it obtained the results comparable to Glove Vectors, trained on 6 billion tokens. This indicates that our ReIn+ReOut is effective even without extra training data (only 783K training tokens in en).

	en
Google News Vectors	258
GloVe Vectors	290
ReIn+ReOut	289

Table 9: Perplexity results of standard LSTM compared with off-the-shelf vectors.

## 6 Conclusion

We proposed a simple and effective method to incorporate word clusters into the CBOW model. Our method is effective on several downstream tasks. For future work, we will test our methods on larger corpora and also add more downstream tasks. We will also study how to combine word clusters and subword information.

## Acknowledgments

We would like to thank anonymous reviewers for their constructive comments and Hu also thanks his support from China Scholarship Council.

## References

Waleed Ammar, George Mulcaire, Yulia Tsvetkov, Guillaume Lample, Chris Dyer, and Noah A Smith.

<sup>8</sup><https://code.google.com/archive/p/word2vec/>

2016. Massively multilingual word embeddings. *arXiv preprint arXiv:1602.01925*.
- Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron C. Courville, and Yoshua Bengio. 2017. [An actor-critic algorithm for sequence prediction](#). In *International Conference on Learning Representations*.
- Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.
- Jan A. Botha, Emily Pitler, Ji Ma, Anton Bakalov, Alex Salcianu, David Weiss, Ryan T. McDonald, and Slav Petrov. 2017. Natural language processing with small feed-forward networks. In *EMNLP*.
- Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. Wit<sup>3</sup>: Web inventory of transcribed and translated talks. In *Proceedings of the 16<sup>th</sup> Conference of the European Association for Machine Translation (EAMT)*, pages 261–268, Trento, Italy.
- Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. 2014. Report on the 11th iwslt evaluation campaign, iwslt 2014. In *Proceedings of the International Workshop on Spoken Language Translation, Hanoi, Vietnam*, page 57.
- Xinxiong Chen, Lei Xu, Zhiyuan Liu, Maosong Sun, and Huanbo Luan. 2015. Joint learning of character and word embeddings. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.
- Jon Dehdari, Liling Tan, and Josef van Genabith. 2016. [BIRA: Improved predictive exchange word clustering](#). In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*, pages 1169–1174, San Diego, CA, USA. Association for Computational Linguistics.
- Daniela Gerz, Ivan Vulić, Edoardo Ponti, Jason Naradowsky, Roi Reichart, and Anna Korhonen. 2018. Language modeling for morphologically rich languages: Character-aware modeling for word-level prediction. *Transactions of the Association of Computational Linguistics*, 6:451–465.
- Po-Sen Huang, Chong Wang, Sitao Huang, Dengyong Zhou, and Li Deng. 2018. [Towards neural phrase-based machine translation](#). In *International Conference on Learning Representations*.
- Lingpeng Kong, Nathan Schneider, Swabha Swayamdipta, Archna Bhatia, Chris Dyer, and Noah A. Smith. 2014. A dependency parser for tweets. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1001–1012.
- Minh-Thang Luong and Christopher D. Manning. 2015. Stanford neural machine translation systems for spoken language domain. In *International Workshop on Spoken Language Translation*.
- Thang Luong, Richard Socher, and Christopher Manning. 2013. Better word representations with recursive neural networks for morphology. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 104–113.
- Stephen Merity, Nitish Shirish Keskar, and Richard Socher. 2018. [Regularizing and optimizing lstm language models](#). In *International Conference on Learning Representations*.
- Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.
- Arvind Neelakantan, Jeevan Shankar, Alexandre Pasos, and Andrew McCallum. 2014. Efficient non-parametric estimation of multiple embeddings per word in vector space. In *EMNLP*.
- Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. [Glove: Global vectors for word representation](#). In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.
- Siyu Qiu, Qing Cui, Jiang Bian, Bin Gao, and Tie-Yan Liu. 2014. Co-learning of word representations and morpheme representations. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 141–150.
- Alan Ritter, Sam Clark, Oren Etzioni, et al. 2011. Named entity recognition in tweets: an experimental study. In *Proceedings of the conference on empirical methods in natural language processing*, pages 1524–1534. Association for Computational Linguistics.
- Raphael Shu and Hideki Nakayama. 2018. [Compressing word embeddings via deep compositional code learning](#). In *International Conference on Learning Representations*.
- Clara Vania and Adam Lopez. 2017. [From characters to words to in between: Do we capture morphology?](#) In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2016–2027, Vancouver, Canada. Association for Computational Linguistics.
- Joern Wuebker, Stephan Peitz, Felix Rietig, and Hermann Ney. 2013. Improving statistical machine translation with word class models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1377–1381.

## A Experiments on Word Similarity Task

The word similarity task is not necessarily suitable for our replacing method due to many infrequent words sharing the same embedding within one cluster. Thus, we report the results of the task for three different groups of test word pairs: *frequent-word-pair* consisting of frequent word pairs, *infrequent-word-pair* consisting of word pairs that share a cluster embedding with other words, and *all-word-pair* consisting of all test word pairs. We used the publicly available enwik8<sup>9</sup> corpus as the training data to obtain both word embeddings and word clusters. Note that we use this data only for the word similarity task, not for downstream tasks such has language modelling and machine translation. We preprocessed the corpus by lowercasing all words, removing words that contain non-alphabetical characters, and removing words whose frequency is less than 5. The final corpus contains approximately 12 million tokens and 60K word types. We chose MEN, MTurk287, MTurk771, RW and WS353 as our datasets. Then, we evaluated the quality of these representations by computing Spearman's rank correlation coefficient. One straightforward method to incorporate word cluster into CBOW model is to average the embeddings of word and its cluster referred as to AvgIn.

	Dataset (#word pairs)	CBOW	ReIn+ ReOut	AvgIn	AvgIn+ ReOut
Frequent word pair	MTurk287 (198)	65.12	66.03	64.22	<b>66.56</b>
	MEN (1296)	65.09	<b>68.74</b>	61.03	63.34
	WS353 (244)	69.51	<b>70.36</b>	63.20	62.00
	RW (169)	49.13	<b>51.57</b>	45.59	48.83
	MTurk771 (530)	54.10	<b>56.83</b>	48.37	51.34
Infrequent word pair	MTurk287 (86)	<b>49.50</b>	34.28	43.83	36.89
	MEN (1686)	<b>46.71</b>	23.58	23.01	26.61
	WS353 (89)	<b>52.12</b>	33.68	32.35	31.08
	RW (828)	<b>31.42</b>	21.49	20.75	20.68
	MTurk771 (237)	<b>50.88</b>	25.55	31.08	31.35
All word pair	MTurk287 (284)	<b>60.98</b>	58.14	58.49	58.42
	MEN (2982)	<b>54.79</b>	44.65	40.55	43.70
	WS353 (333)	<b>64.41</b>	58.61	53.26	52.96
	RW (997)	<b>35.15</b>	25.12	23.92	24.78
	MTurk771 (767)	<b>52.73</b>	46.93	42.50	45.10

Table 10: Spearman's rank correlation coefficient on word similarity datasets for different groups. The best scores in each group are in bold.

We first applied ClusterCat to the preprocessed corpus to obtain word clusters and then produced cluster-incorporated word embeddings with ReIn+ReOut. The results are shown in Table 10. In ReIn+ReOut, the number of input words is 10,203, which is the sum of 9,603 frequent words and 600 clusters. This is only 16.9% of the number of in-

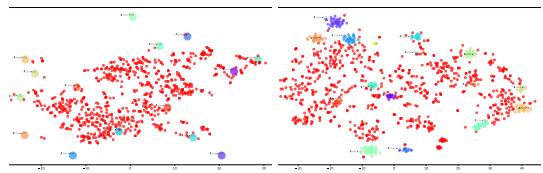


Figure 2: Visualization of the embeddings of frequent words and clusters before fine-tuning (left) and the embeddings of frequent and infrequent words after fine-tuning (right). The red circle represents frequent words. The color of infrequent words within different clusters are different (right), and the big circle represents word clusters (left).

put words for the original CBOW, which maintains 60K input words. In all word pair group, CBOW outperformed ReIn+ReOut on all datasets. This is because ReIn+ReOut does not perform well in infrequent-word-pair group as many infrequent words share exactly the same embedding in one cluster. In this experiment, each cluster had 82 words on average. However, ReIn+ReOut outperformed CBOW on frequent-word-pair group in all datasets. This result suggests that ReIn+ReOut is effective in learning embeddings for frequent words with much fewer parameters. AvgIn underperformed CBOW in all-word-pair group, which suggests that this straightforward way to incorporate word clusters is not effective. We also found that AvgIn+ReOut can improve the performance on 3 datasets in all-word-pairs group compared with AvgIn. However, AvgIn+ReOut still underperformed CBOW on all datasets.

## B Visualization of Word Embeddings

We visualize word embeddings using t-SNE projections. Specifically, we randomly chose 15 clusters and all frequent words from en and visualize frequent and infrequent word embeddings in these 15 clusters in Figure 2. The embeddings of infrequent words within one cluster are located close together after being fine-tuned. Some infrequent word embeddings are updated only several times and are not far away from where they were randomly initialized, and now they become more reliable.

<sup>9</sup><http://mattmahoney.net/dc/enwik8.zip>

# Investigating Learning Dynamics of BERT Fine-Tuning

Yaru Hao<sup>†\*</sup>, Li Dong<sup>‡</sup>, Furu Wei<sup>‡</sup>, Ke Xu<sup>†</sup>

<sup>†</sup>Beihang University

<sup>‡</sup>Microsoft Research

{haoyaru@, kexu@nlsde.}buaa.edu.cn  
{lidong1, fuwei}@microsoft.com

## Abstract

The recently introduced pre-trained language model BERT advances the state-of-the-art on many NLP tasks through the fine-tuning approach, but few studies investigate how the fine-tuning process improves the model performance on downstream tasks. In this paper, we inspect the learning dynamics of BERT fine-tuning with two indicators. We use JS divergence to detect the change of the attention mode and use SVCCA distance to examine the change to the feature extraction mode during BERT fine-tuning. We conclude that BERT fine-tuning mainly changes the attention mode of the last layers and modifies the feature extraction mode of the intermediate and last layers. Moreover, we analyze the consistency of BERT fine-tuning between different random seeds and different datasets. In summary, we provide a distinctive understanding of the learning dynamics of BERT fine-tuning, which sheds some light on improving the fine-tuning results.

## 1 Introduction

BERT (Bidirectional Encoder Representations from Transformers; Devlin et al. 2019) is a large pre-trained language model. It obtains state-of-the-art results on a wide array of Natural Language Processing (NLP) tasks. Unlike other previous pre-trained language models (Peters et al., 2018a; Radford et al., 2018), BERT employs the multi-layer bidirectional Transformer encoder as the model architecture and proposes two novel pre-training tasks: the masked language modeling and the next sentence prediction.

There are two approaches to adapt the pre-trained language representations to the downstream tasks. One is the feature-based approach, where the parameters of the original pre-trained

model are frozen when applied on the downstream tasks (Mikolov et al., 2013; Pennington et al., 2014; Peters et al., 2018a). Another one is the fine-tuning approach, where the pre-trained model and the task-specific model are trained together (Dai and Le, 2015; Howard and Ruder, 2018; Radford et al., 2018). Take the classification task as an example, the new parameter added for BERT fine-tuning is a task-specific fully-connected layer, then all parameters of BERT and the classification layer are trained together to minimize the loss function.

Peters et al. (2019) demonstrate that the fine-tuning approach of BERT generally outperforms the feature-based approach. We know that BERT encodes task-specific representations during fine-tuning, but it is unclear about the learning dynamics of BERT fine-tuning, i.e., how fine-tuning helps BERT to improve performance on downstream tasks.

We investigate the learning dynamics of BERT fine-tuning with two indicators. First, we use Jensen-Shannon divergence to measure the change of the attention mode during BERT fine-tuning. Second, we use Singular Vector Canonical Correlation Analysis (SVCCA; Raghu et al. (2017)) distance to measure the change of the feature extraction mode.

We conclude that during the fine-tuning procedure, BERT mainly changes the attention mode of the last layers, and modifies the feature extraction mode of intermediate and last layers. At the same time, BERT has the ability to avoid catastrophic forgetting of knowledge in low layers. Moreover, we also analyze the consistency of the fine-tuning procedure. Across different random seeds and different datasets, we observe that the changes of low layers (0-9th layer) are generally consistent, which indicates that BERT has learned some common transferable language knowledge in low layers during the pre-training process, while the task-specific

\*Contribution during internship at Microsoft Research.

information is mostly encoded in intermediate and last layers.

## 2 Experimental Setup

We employ the BERT-large model<sup>1</sup> on a diverse set of NLP tasks: natural language inference (NLI), sentiment analysis (SA) and paraphrase detection (PD).

For NLI, we use both the Multi-Genre Natural Language Inference dataset (MNLI; Williams et al. 2018) and the Recognizing Textual Entailment dataset (RTE; aggregated from Dagan et al. 2006, Haim et al. 2006, Giampiccolo et al. 2007, Bentivogli et al. 2009). For SA, we use the binary version of the Stanford Sentiment Treebank dataset (SST-2; Socher et al. 2013). For PD, we use the Microsoft Research Paraphrase Corpus dataset (MRPC; Dolan and Brockett 2005).

Dataset	LR	BS	NE
MNLI	3e-5	64	3
RTE	1e-5	32	5
SST-2	3e-5	64	4
MRPC	1e-5	16	5

Table 1: Hyperparameter configuration for BERT fine-tuning. LR: learning rate, BS: batch size, NE: number of epochs.

The hyperparameter choice for fine-tuning is task-specific. We choose relatively optimal parameters for every dataset as suggested in Devlin et al. (2019). The detailed hyperparameter configuration is shown in Table 1. Moreover, we use Adam optimizer with the slanted triangular learning rate schedule (Howard and Ruder, 2018) and keep the dropout probability at 0.1.

## 3 Fine-tuning changes the attention mode of the last layers

The model architecture of BERT is essentially based on the multi-layer bidirectional Transformer, the core function of which is the self-attention mechanism (Vaswani et al., 2017). We use Jensen-Shannon divergence between two attention scores to detect changes of the attention mode in different layers during fine-tuning.

**Jensen-Shannon divergence** JS divergence is a method of measuring the distance between two

probability distributions, it is defined as:

$$D_{JS}(P||Q) = \frac{1}{2}D_{KL}(P||R) + \frac{1}{2}D_{KL}(Q||R)$$

where  $P$  and  $Q$  are two different probability distributions,  $R = \frac{P+Q}{2}$  is the average probability distribution of them and  $D_{KL}$  represents the Kullback-Leibler divergence.

For every layer of BERT, there are 16 attention heads, each head produces an attention score of the input sequence. Each attention score is a probability distribution about how much attention a target word pays to other words. We compute JS divergence of attention scores between the original BERT model  $M_0$  and the fine-tuned model  $M_t$  on the development set, by calculating the average of the sum of JS divergence at each word and each attention head for every layer, the specific calculation formula is as follows:

$$D_{JS}(M_t||M_0) = \frac{1}{N} \frac{1}{H} \sum_{n=1}^N \sum_{h=1}^H \frac{1}{W} \sum_{i=1}^W D_{JS}(A_t^h(word_i)||A_0^h(word_i))$$

where  $N$  denotes the number of development examples,  $H$  denotes the number of attention heads,  $W$  denotes the number of tokens in a sequence and  $A_t^h(word_i)$  denotes the attention score of the attention head  $h$  at  $word_i$  in model  $M_t$ .

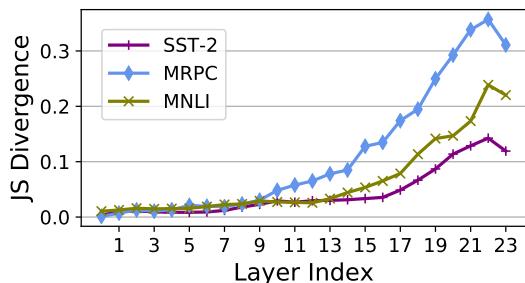


Figure 1: JS divergence of attention scores of every layer between the original BERT model and the fine-tuned model.

We present JS divergence results in Figure 1, from which we observe the attention mode in low layers and intermediate layers do not change seriously, while the attention mode of last layers changes drastically. It indicates that the fine-tuning procedure has the ability to keep the attention mode of low layers consistent with the original BERT model, and changes the attention mode of the last layers to adapt BERT on specific tasks.

<sup>1</sup>[github.com/google-research/bert](https://github.com/google-research/bert)

## 4 Fine-tuning modifies the feature extraction mode of the intermediate and the last layers

While the attention score implies the inherent dependencies between different words, the output representation of every layer is the practical feature that the model extracts. We use SVCCA distance (Raghu et al., 2017) to quantify the change of these output representations during fine-tuning, which indicates the change of the feature extraction mode of BERT.

### Singular Vector Canonical Correlation Analysis.

SVCCA distance is used as a metric to measure the differences of hidden representations between the original BERT model  $M_0$  and the fine-tuned model  $M_t$  at a target layer. It is calculated by:

$$D_{SVCCA}(M_t || M_0) = 1 - \frac{1}{c} \sum_{i=1}^c \rho^{(i)}$$

where  $c$  denotes the hidden size of BERT,  $\rho$  is the Canonical Correlation Analysis (CCA) resulting in a value between 0 and 1, which indicates how well correlated the two representations derived by two models are. For a detailed explanation of SVCCA, please see Raghu et al. (2017).

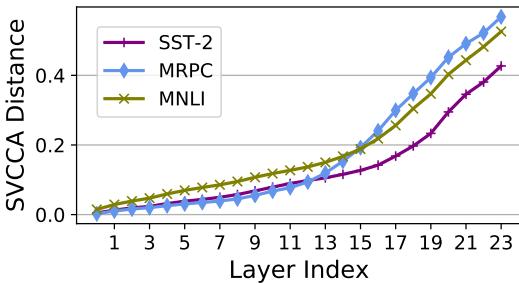


Figure 2: SVCCA distance of individual layers between the original BERT model and the fine-tuned model.

From Figure 2, we observe that changes in SVCCA distance in higher layers are more distinct than lower layers. This phenomenon is reasonable because the output representation of higher layers undergoes more transformations, so the change of SVCCA distance in higher layers is more dramatic.

As the output representation of the last layer is directly used for classification, we aim to compare the effect of each layer on the final output representation respectively. We replace the parameters of

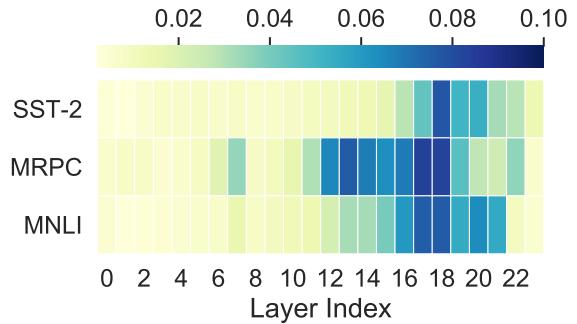


Figure 3: SVCCA distance of the last layer between the original fine-tuned model and the fine-tuned model with parameters of a target layer replaced with their pre-trained values.

every layer in the fine-tuned model with their original values in the BERT model before fine-tuning and compute the SVCCA distance of the last layer output representation. The results are shown in Figure 3, we observe that whether the low layers (0-10) are replaced with their original values or not, it has little effect on the final output representation. Moreover, the change in the intermediate and last layers will increase the SVCCA distance, which reflects that fine-tuning mainly changes the feature extraction mode of intermediate and last layers.

## 5 Consistency of Fine-tuning

In this section, we investigate the consistency of different fine-tuning procedures, including the consistency between different random seeds and the consistency between different datasets.

### 5.1 Consistency between different random seeds

We fine-tune two models on every dataset with the same hyperparameters but different random seeds. We compute the pairwise JS divergence and SVCCA distance of each layer between the two models with different random seeds.

As shown in Figure 4, for large dataset MNLI and SST-2, the attention mode of low and intermediate layers is basically consistent between two different random seeds, whereas the attention mode of last layers is relatively divergent. For MRPC, the attention mode appears to be divergent at the 9th layer.

Figure 5 illustrates SVCCA distance between different random seeds, we observe that the SVCCA distance gradually increases in all layers. For MNLI and SST-2, the increase of last layers is

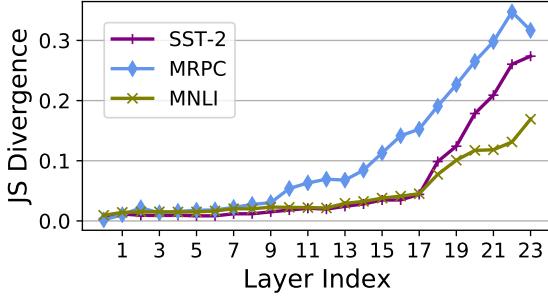


Figure 4: JS divergence between two models with different random seeds.

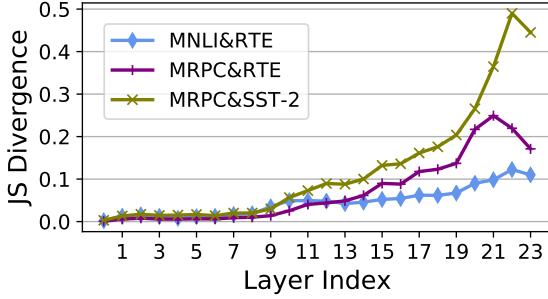


Figure 6: JS divergence between different datasets.

more obvious, and for MRPC, the increase appears to be obvious from the 13th layer.

## 5.2 Consistency between different datasets

Besides the consistency between different random seeds, we also aim to investigate the consistency between different datasets. We fine-tune two models on two different datasets then evaluate on a combined dataset containing 200 examples respectively from both two datasets.

For different datasets of the same domain, we use two models fine-tuned on RTE and MNLI dataset. For different domains, we examine the consistency between MRPC and RTE, which both have pairwise input sequences, and the consistency between MRPC and SST-2, which have different patterns of input sequences. The JS divergence results and SVCCA distance results between different datasets are shown in Figure 6 and Figure 7.

Figure 6 and Figure 7 demonstrate that no matter two datasets are from the same domain or the different domain, the attention mode and the feature extraction mode of low layers (0-7 layer) are consistent, which indicates BERT studies some common language knowledge during the pre-training procedure and low layers are stable to change their original modes. JS divergence of the attention scores

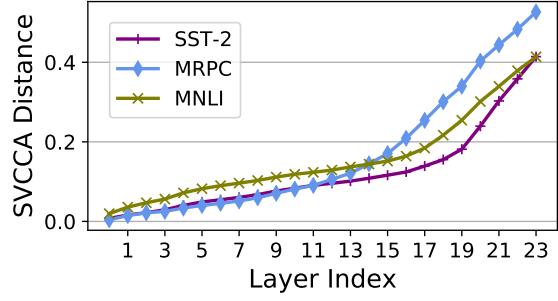


Figure 5: SVCCA distance between two models with different random seeds.

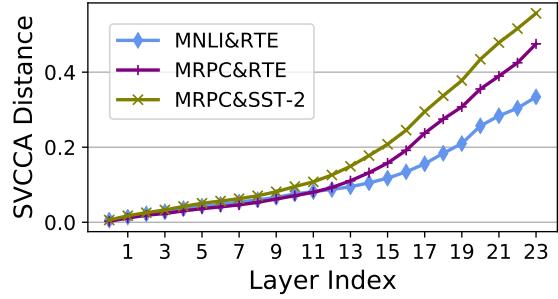


Figure 7: SVCCA distance between different datasets.

and SVCCA distance of the output representations in intermediate and last layers between two models are more distinct when the difference between two training datasets increases. The consistency between datasets from similar tasks like RTE and MNLI is still relatively strong in last layers compared to the consistency between datasets from the different domain. And when the input sequence pattern and the domain of two datasets are different, the consistency of intermediate and last layers is weak as expected.

## 6 Related Work

Pre-trained language models (Radford et al., 2018; Devlin et al., 2019; Liu et al., 2019; Dong et al., 2019; Yang et al., 2019; Clark et al., 2020; Bao et al., 2020) stimulate the research interest on the interpretation of these black-box models. Peters et al. (2018b) show that the biLM-based models learn representations that vary with network depth, the lower layers specialize in local syntactic relationships and the higher layers model longer range relationships. Kovaleva et al. (2019) propose a methodology and offer the analysis of BERTs capacity to capture different kinds of linguistic information by encoding it in its self-attention weights. Hao et al. (2019) visualize the loss landscapes and

optimization trajectories of the BERT fine-tuning procedure and find that low layers of the BERT model are more invariant and transferable across tasks. Merchant et al. (2020) find that fine-tuning primarily affects the top layers of BERT, but with noteworthy variation across tasks. Hao et al. (2020) propose a self-attention attribution method to interpret information flow within Transformer.

## 7 Discussions

We use JS divergence to detect the change of the attention mode in different layers during BERT fine-tuning and use SVCCA distance to detect the change of the feature extraction mode. We observe that BERT fine-tuning mainly changes the attention mode of last layers and modifies the feature extraction mode of intermediate and last layers.

We also demonstrate that the changes of low layers are consistent between different random seeds and different datasets, which indicates that BERT learns common transferable language knowledge in low layers. In future research, we would like to explore learning dynamics for cross-lingual pre-trained models (Conneau and Lample, 2019; Conneau et al., 2020; Chi et al., 2020).

## Acknowledgements

The work was partially supported by National Natural Science Foundation of China (NSFC) [Grant No. 61421003].

## References

- Hangbo Bao, Li Dong, Furu Wei, Wenhui Wang, Nan Yang, Xiaodong Liu, Yu Wang, Songhao Piao, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2020. UniLMv2: Pseudo-masked language models for unified language model pre-training. *arXiv preprint arXiv:2002.12804*.
- Luisa Bentivogli, Ido Dagan, Hoa Trang Dang, Danilo Giampiccolo, and Bernardo Magnini. 2009. The fifth pascal recognizing textual entailment challenge. In *In Proc Text Analysis Conference (TAC09)*.
- Zewen Chi, Li Dong, Furu Wei, Nan Yang, Saksham Singhal, Wenhui Wang, Xia Song, Xian-Ling Mao, Heyan Huang, and Ming Zhou. 2020. Infoxlm: An information-theoretic framework for cross-lingual language model pre-training. *CoRR*, abs/2007.07834.
- Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *ICLR*.
- Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. *Unsupervised cross-lingual representation learning at scale*. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.
- Alexis Conneau and Guillaume Lample. 2019. *Cross-lingual language model pretraining*. In *Advances in Neural Information Processing Systems*, pages 7057–7067. Curran Associates, Inc.
- Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. *The pascal recognising textual entailment challenge*. In *Machine learning challenges. evaluating predictive uncertainty, visual object classification, and recognising textual entailment*, pages 177–190.
- Andrew M. Dai and Quoc V. Le. 2015. *Semi-supervised sequence learning*. *CoRR*, abs/1511.01432.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. *BERT: Pre-training of deep bidirectional transformers for language understanding*. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- William B. Dolan and Chris Brockett. 2005. *Automatically constructing a corpus of sentential paraphrases*. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.
- Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*.
- Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. 2007. *The third PASCAL recognizing textual entailment challenge*. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 1–9, Prague. Association for Computational Linguistics.
- Roy Bar Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. 2006. *The second pascal recognising textual entailment challenge*. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*.
- Yaru Hao, Li Dong, Furu Wei, and Ke Xu. 2019. *Visualizing and understanding the effectiveness of BERT*. In *Proceedings of the 2019 Conference on Empirical*

- Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4143–4152, Hong Kong, China. Association for Computational Linguistics.
- Yaru Hao, Li Dong, Furu Wei, and Ke Xu. 2020. *Self-attention attribution: Interpreting information interactions inside transformer*. *CoRR*, abs/2004.11207.
- Jeremy Howard and Sebastian Ruder. 2018. *Fine-tuned language models for text classification*. *CoRR*, abs/1801.06146.
- Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. *Revealing the dark secrets of BERT*. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4365–4374, Hong Kong, China. Association for Computational Linguistics.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.
- Amil Merchant, Elahe Rahimtoroghi, Ellie Pavlick, and Ian Tenney. 2020. *What happens to bert embeddings during fine-tuning?*
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. *Distributed representations of words and phrases and their compositionality*. *CoRR*, abs/1310.4546.
- Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. *Glove: Global vectors for word representation*. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.
- Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018a. *Deep contextualized word representations*. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.
- Matthew Peters, Mark Neumann, Luke Zettlemoyer, and Wen-tau Yih. 2018b. *Dissecting contextual word embeddings: Architecture and representation*. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1499–1509, Brussels, Belgium. Association for Computational Linguistics.
- Matthew E. Peters, Sebastian Ruder, and Noah A. Smith. 2019. *To tune or not to tune? adapting pre-trained representations to diverse tasks*. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 7–14, Florence, Italy. Association for Computational Linguistics.
- Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. *Improving language understanding by generative pre-training*.
- Maithra Raghu, Justin Gilmer, Jason Yosinski, and Jascha Sohl-Dickstein. 2017. *Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability*. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6076–6085. Curran Associates, Inc.
- Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. *Recursive deep models for semantic compositionality over a sentiment treebank*. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. *Attention is all you need*. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.
- Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. *A broad-coverage challenge corpus for sentence understanding through inference*. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.
- Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019. XLNet: Generalized autoregressive pretraining for language understanding. In *33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*.

# Second-Order Neural Dependency Parsing with Message Passing and End-to-End Training

Xinyu Wang and Kewei Tu\*

School of Information Science and Technology, ShanghaiTech University

Shanghai Engineering Research Center of Intelligent Vision and Imaging

Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Sciences

University of Chinese Academy of Sciences

{wangxyl,tukw}@shanghaitech.edu.cn

## Abstract

In this paper, we propose second-order graph-based neural dependency parsing using message passing and end-to-end neural networks. We empirically show that our approaches match the accuracy of very recent state-of-the-art second-order graph-based neural dependency parsers and have significantly faster speed in both training and testing. We also empirically show the advantage of second-order parsing over first-order parsing and observe that the usefulness of the head-selection structured constraint vanishes when using BERT embedding.

## 1 Introduction

Graph-based dependency parsing is a popular approach to dependency parsing that scores parse components of a sentence and then finds the highest scoring tree through inference. First-order graph-based dependency parsing takes individual dependency edges as the components of a parse tree, while higher-order dependency parsing considers more complex components consisting of multiple edges. There exist both exact inference algorithms (Carreras, 2007; Koo and Collins, 2010; Ma and Zhao, 2012) and approximate inference algorithms (McDonald and Pereira, 2006; Smith and Eisner, 2008; Gormley et al., 2015) to find the best parse tree. Recent work focused on neural network based graph dependency parsers (Kiperwasser and Goldberg, 2016; Wang and Chang, 2016; Cheng et al., 2016; Kuncoro et al., 2016; Ma and Hovy, 2017; Dozat and Manning, 2017). Dozat and Manning (2017) proposed a first-order graph-based neural dependency parsing approach with a simple head-selection training objective. It uses a biaffine function to score dependency edges and has high efficiency and good performance. Subsequent work

introduced second-order inference into their parser. Ji et al. (2019) proposed a graph neural network that captures second-order information in token representations, which are then used for first-order parsing. Very recently, Zhang et al. (2020) proposed an efficient second-order tree CRF model for dependency parsing and achieved state-of-the-art performance.

In this paper, we first show how a previously proposed second-order semantic dependency parser (Wang et al., 2019) can be applied to syntactic dependency parsing with simple modifications. The parser is an end-to-end neural network derived from message passing inference on a conditional random field that encodes the second-order parsing problem. We then propose an alternative conditional random field that incorporates the head-selection constraint of syntactic dependency parsing, and derive a novel second-order dependency parser. We empirically compare the two second-order approaches and the first-order baselines on English Penn Tree Bank 3.0 (PTB), Chinese Penn Tree Bank 5.1 (CTB) and datasets of 12 languages in Universal Dependencies (UD). We show that our approaches achieve state-of-the-art performance on both PTB and CTB and our approaches are significantly faster than recently proposed second-order parsers.

We also make two interesting observations from our empirical study. First, it is a common belief that contextual word embeddings such as ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019) already conveys sufficient high-order information that renders high-order parsing less useful, but we find that second-order decoding is still helpful even with strong contextual embeddings like BERT. Second, while Zhang et al. (2019) previously found that incorporating the head-selection constraint is helpful in first-order parsing, we find that with a better loss function design and hyper-parameter tun-

\* Kewei Tu is the corresponding author.

ing both first- and second-order parsers without the head-selection constraint can match the accuracy of parsers with the head-selection constraint and can even outperform the latter when using BERT embedding.

Our approaches are closely related to the work of Gormley et al. (2015), which proposed a non-neural second-order parser based on Loopy Belief Propagation (LBP). Our work differs from theirs in that: 1) we use Mean Field Variational Inference (MFVI) instead of LBP, which Wang et al. (2019) found is faster and equally accurate in practice; 2) we add the head-selection constraint and do not include the global tree constraint that is shown to produce only slight improvement (Zhang et al., 2019) but would complicate our neural network design and implementation; 3) we employ modern neural encoders and achieve much better parsing accuracy. Our approaches are also closely related to the very recent work of Fonseca and Martins (2020). The main difference is that we use MFVI while they use the dual decomposition algorithm AD<sup>3</sup> (Martins et al., 2011, 2013) for approximate inference.

## 2 Approach

Zhang et al. (2019) categorized different kinds of graph-based dependency parsers based on their structured output constraints according to the normalization for output scores. A **Local** approach views dependency parsing as a head-selection problem, in which each word selects exactly one dependency head. A **Single** approach places no structured constraint, viewing the existence of each possible dependency edge as an independent binary classification problem.

The second-order semantic dependency parser of Wang et al. (2019) is an end-to-end neural network derived from message passing inference on a conditional random field that encodes the second-order parsing problem. It is clearly a **Single** approach because of the lack of structured constraints in semantic dependency parsing. We can apply this approach to syntactic dependency parsing with two minor modifications. First, co-parents, one of the three types of second-order parts, become invalid and hence are removed. Second, for the approach to output valid parse trees during testing, we run maximum spanning tree (MST) (McDonald et al., 2005) based on the posterior edge probabilities predicted by the approach.

Inspired by Wang et al. (2019), below we propose a **Local** second-order parsing approach. While the **Single** approach uses Boolean random variables to represent existence of possible dependency edges, our **Local** approach defines a discrete random variable for each word specifying its dependency head, thus enforcing the head-selection constraint and leading to different formulation of the message passing inference steps.

### 2.1 Scoring

Following Dozat and Manning (2017), we predict edge existence and edge labels separately. Suppose the input sentence is  $\mathbf{w} = [w_0, w_1, w_2, \dots, w_n]$  where  $w_0$  is a dummy root. We feed word representations outputted by the BiLSTM encoder into a biaffine function to assign score  $s_{ij}^{(\text{edge})}$  to edge  $w_i \rightarrow w_j$ . We use a Trilinear function to assign score  $s_{ij,ik}^{(\text{sib})}$  to the siblings part consisting of edges  $w_i \rightarrow w_j$  and  $w_i \rightarrow w_k$ , and another Trilinear function to assign score  $s_{ij,jk}^{(\text{gp})}$  to the grandparent part consisting of edges  $w_i \rightarrow w_j$  and  $w_j \rightarrow w_k$ . For edge labels, we use a biaffine function to predict label scores of each potential edge and use a softmax function to compute the label distribution  $P(y_{ij}^{(\text{label})} | \mathbf{w})$ , where  $y_{ij}^{(\text{label})}$  represents the possible label for edge  $w_i \rightarrow w_j$ .

### 2.2 Message Passing

The head-selection structured constraint requires that each word except the root has exactly one head. We define variable  $X_j \in \{0, 1, 2, \dots, n\}$  to indicate the head of word  $w_j$ . We then define a conditional random field (CRF) over  $[X_1, \dots, X_n]$ . For each variable  $X_j$ , the unary potential is defined by:

$$\phi_u(X_j = i) = \exp(s_{ij}^{(\text{edge})})$$

Given two variables  $X_j$  and  $X_l$ , the binary potential is defined by:

$$\phi_p(X_j = i, X_l = k) = \begin{cases} \exp(s_{ij,kl}^{(\text{sib})}) & k = i \\ \exp(s_{ij,kl}^{(\text{gp})}) & k = j \\ 1 & \text{Otherwise} \end{cases}$$

We use MFVI for approximate inference on this CRF. The algorithm updates the factorized poste-

prior distribution  $Q_j(X_j)$  of each word iteratively.

$$\begin{aligned}\mathcal{M}_j^{(t-1)}(i) &= \sum_{k \neq i,j} Q_k^{(t-1)}(i) s_{ij,ik}^{(sib)} \\ &\quad + Q_k^{(t-1)}(j) s_{ij,jk}^{(gp)} + Q_i^{(t-1)}(k) s_{ki,ij}^{(gp)} \\ Q_j^{(t)}(i) &= \frac{\exp\{s_{ij}^{(\text{edge})} + \mathcal{M}_j^{(t-1)}(i)\}}{\sum_{k=0}^n \exp\{s_{kj}^{(\text{edge})} + \mathcal{M}_j^{(t-1)}(k)\}}\end{aligned}$$

At  $t = 0$ ,  $Q_j^{(t)}(X_j)$  is initialized by normalizing the unary potential. The iterative update steps can be unfolded as recurrent neural network layers parameterized by part scores, thus forming an end-to-end neural network.

Compared with the update formula in the **Single** approach, here the posterior distributions are defined over head-selections and are normalized over all possible heads. The computational complexity remains the same.

### 2.3 Learning

We define the cross entropy losses by:

$$\begin{aligned}\mathcal{L}^{(\text{edge})} &= - \sum_i \log[Q_i(y_i^{*(\text{edge})} | \mathbf{w})] \\ \mathcal{L}^{(\text{label})} &= - \sum_{i,j} \mathbb{1}(y_j^{*(\text{edge})} = i) \log(P(y_{ij}^{*(\text{label})} | \mathbf{w})) \\ \mathcal{L} &= \lambda \mathcal{L}^{(\text{label})} + (1 - \lambda) \mathcal{L}^{(\text{edge})}\end{aligned}$$

where  $y_i^{*(\text{edge})}$  is the head of word  $w_i$  and  $y_{ij}^{*(\text{label})}$  is the label of edge  $w_i \rightarrow w_j$  in the golden parse tree,  $\lambda$  is a hyper-parameter and  $\mathbb{1}(x)$  is an indicator function that returns 1 when  $x$  is true and 0 otherwise.

## 3 Experiments

### 3.1 Setups

Following previous work (Dozat and Manning, 2017; Ma et al., 2018), we use PTB 3.0 (Marcus et al., 1993), CTB 5.1 (Xue et al., 2002) and 12 languages in Universal Dependencies (Nivre et al., 2018) (UD) 2.2 to evaluate our parser. Punctuation is ignored in all the evaluations. We use the same treebanks and preprocessing as Ma et al. (2018) for PTB, CTB, and UD. For all the datasets, we remove sentences longer than 90 words in training sets for faster computation.

We use **GNN**, **Local1O**, **Single1O**, **Local2O** and **Single2O** to represent the approaches of Ji et al. (2019), Dozat and Manning (2017), Dozat

Hidden Layer	Hidden Sizes
Word/GloVe/Char	100
POS	50
GloVe Linear	125
BERT Linear	125
BiLSTM	3*600
Char LSTM	1*400
Unary Arc (UD)	500
<b>Local1O/Local2O</b>	Unary Arc (Others)
<b>Single1O/Single2O</b>	Unary Arc (Others)
Label	150
Binary Arc	150
Dropouts	Dropout Prob.
Word/GloVe/POS	20%
Char LSTM (FF/recur)	33%
Char Linear	33%
BiLSTM (FF/recur)	45%/25%
Unary Arc/Label	25%/33%
Binary Arc	25%
Optimizer & Loss	Value
<b>Local1O/Local2O</b>	Interpolation ( $\lambda$ )
<b>Single1O/Single2O</b>	Interpolation ( $\lambda$ )
Adam $\beta_1$	0
Adam $\beta_2$	0.95
Decay Rate	0.85
Decay Step (without dev improvement)	500
Weight Initialization	Mean/Stddev
Unary weight	0.0/1.0
Binary weight	0.0/0.25

Table 1: Hyper-parameter for **Local1O**, **Single2O** and **Local2O** in our experiment.

and Manning (2018), and our two second-order approaches respectively. For all the approaches, we use the MST algorithm to guarantee tree-structured output in testing. We use the concatenation of word embeddings, character-level embeddings and part-of-speech (POS) tag embeddings to represent words and additionally concatenate BERT embeddings for experiments with BERT. For a fair comparison with previous work, we use GloVe (Pennington et al., 2014) and BERT-Large-Uncased model for PTB, and structured-skipgram (Ling et al., 2015) and BERT-Base-Chinese model for CTB. For UD, we use fastText embeddings (Bojanowski et al., 2017) and BERT-Base-Multilingual-Cased model for different languages. We set the default iteration number for our approaches to 3 because we find no improvement on more or less iterations.

For **GNN**<sup>1</sup>, we rerun the code based on the official release of Ji et al. (2019). For **Single1O**, **Local1O**<sup>2</sup>, **Single2O**<sup>3</sup>, we implement these ap-

<sup>1</sup><https://github.com/AntNLP/gnn-dep-parsing>

<sup>2</sup><https://github.com/tdozat/Parser-v3>

<sup>3</sup>[https://github.com/wangxinyu0922/Second\\_Order\\_SDP](https://github.com/wangxinyu0922/Second_Order_SDP)

	PTB		CTB	
	UAS	LAS	UAS	LAS
Dozat and Manning (2017)	95.74	94.08	89.30	88.23
Ma et al. (2018) <sup>♦</sup>	95.87	94.19	90.59	89.29
F&G (2019) <sup>♦</sup>	96.04	94.43	-	-
GNN	95.87	94.15	90.78	89.50
Single1O	95.75	94.04	90.53	89.28
Local1O	95.83	94.23	90.59	89.28
Single2O	95.86	94.19	90.75	89.55
Local2O	95.98	94.34	<b>90.81</b>	<b>89.57</b>
Ji et al. (2019) <sup>†</sup>	95.97	94.31	-	-
Zhang et al. (2020) <sup>†‡</sup>	<b>96.14</b>	<b>94.49</b>	-	-
Local2O <sup>†‡</sup>	96.12	94.47	-	-
<b>+BERT</b>				
Zhou and Zhao (2019) <sup>♣</sup>	97.20	95.72	-	-
Clark et al. (2018) <sup>◦</sup>	96.60	95.00	-	-
Single1O	96.82	95.20	92.73	91.64
Local1O	96.86	95.32	92.47	91.30
Single2O	96.86	95.31	<b>92.78</b>	<b>91.69</b>
Local2O	<b>96.91</b>	<b>95.34</b>	92.55	91.38

Table 2: Comparison of our approaches and the previous state-of-the-art approaches on PTB and CTB. We report our results averaged over 5 runs. <sup>†</sup>: These approaches perform model selection based on the score on the development set. <sup>‡</sup>: These approaches do not use POS tags as input. <sup>◦</sup>: Clark et al. (2018) uses semi-supervised multi-task learning with ELMo embeddings. <sup>♣</sup>: These approaches use structured-skipgram embeddings instead of GloVe embeddings for PTB. <sup>♣</sup>: For reference, Zhou and Zhao (2019) utilized both dependency and constituency information in their approach. Therefore, the results are not comparable to our results.

proaches based on the official release code of Wang et al. (2019) and we implement **Local2O** based on this code. In speed comparison, we implement the second-order approaches based on an PyTorch implementation biaffine parser<sup>4</sup> implemented by Zhang et al. (2020) for a fair speed comparison with their approach<sup>5</sup>. Since we find that the accuracy of our approaches based on PyTorch implementation on PTB does not change, we only report scores based on Wang et al. (2019).

### 3.2 Hyper-parameters

The hyper-parameters we used in our experiments is shown in Table 1. We tune the the hidden size for calculating  $s_{ij}^{(\text{edge})}$  (Unary Arc in the table) separately for PTB and CTB. Following Qi et al. (2018), we switch to AMSGrad (Reddi et al., 2018) after 5,000 iterations without improvement. We train models for 75,000 iterations with batch sizes of

<sup>4</sup><https://github.com/yzhangcs/parser>

<sup>5</sup>At the time we finished the paper, the official code for the second-order tree CRF parser have not release yet. We believe it is a fair comparison since we use the same settings and GPU as Zhang et al. (2020).

6000 tokens and stopped the training early after 10,000 iterations without improvements on development sets. Different from previous approaches such as Dozat and Manning (2017) and Ji et al. (2019), we use Adam (Kingma and Ba, 2015) with a learning rate of 0.01 and anneal the learning rate by 0.85 for every 500 iterations without improvement on the development set for optimization. For GNN, we train the models with the same setting as in Ji et al. (2019). We do not use character embeddings and our optimization settings for GNN because we find they do not improve the accuracy.

For the edge loss of **Single** approaches, Zhang et al. (2019) proposed to sample a subset of the negative edges to balance positive and negative examples, but we find that using a relatively small interpolation  $\lambda$  (shown in Table 1) on label loss can improve the accuracy and the sampling does not help further improve the accuracy.

### 3.3 Results

Table 2 shows the Unlabeled Attachment Score (UAS) and Labeled Attachment Score (LAS) of all the approaches as well as the reported scores of previous state-of-the-art approaches on PTB and CTB. It can be seen that without BERT, our **Local2O** achieves state-of-the-art performance on CTB and has almost the same accuracy as the very recent work of Zhang et al. (2020) on PTB. With BERT embeddings, **Local2O** performs the best on PTB while **Single2O** has the best accuracy on CTB.

Table 3 shows the results of the five approaches on UD in addition to PTB and CTB. We make the following observations. First, our second-order approaches outperform GNN and the first-order approaches both with and without BERT embeddings, showing that second-order decoders are still helpful in neural parsing even with strong contextual embeddings. Second, without BERT, **Local** slightly outperforms **Single**, although the difference between the two is quite small<sup>6</sup>; when BERT is used, however, **Single** clearly outperforms **Local**, which is quite interesting and warrants further investigation in the future. Third, the relative strength of **Local** and **Single** approaches varies over treebanks, suggesting varying importance of the head-selection constraint.

<sup>6</sup>Note that Zhang et al. (2019) reports higher difference in accuracy between first-order **Local** and **Single** approaches. The discrepancy is most likely caused by our better designed loss function and tuned hyper-parameters.

	<b>PTB</b>	<b>CTB</b>	<b>bg</b>	<b>ca</b>	<b>cs</b>	<b>de</b>	<b>en</b>	<b>es</b>	<b>fr</b>	<b>it</b>	<b>nl</b>	<b>no</b>	<b>ro</b>	<b>ru</b>	<b>Avg.</b>
GNN	94.15	89.50 <sup>†</sup>	90.33	92.39	90.95	79.73	88.43	91.56	87.23	92.44	88.57	89.38	85.26	91.20	89.37
Single1O	94.04	89.28	90.05	92.72 <sup>†</sup>	92.07	81.73	89.55	92.10	88.27	92.64	89.57	91.81	85.39	92.60	90.13
Local1O	94.23	89.28	90.30	92.56	<b>92.15</b>	81.42	89.43	91.99	88.26	92.49	89.76	<b>91.91</b>	85.27	<b>92.72</b>	90.13
Single2O	94.19	89.55 <sup>†</sup>	90.24	92.82 <sup>†</sup>	92.13	<b>81.99<sup>†</sup></b>	89.64 <sup>†</sup>	<b>92.17<sup>†</sup></b>	<b>88.69</b>	<b>92.83<sup>†</sup></b>	89.97 <sup>†</sup>	91.90	85.53 <sup>†</sup>	92.58	90.30 <sup>†</sup>
Local2O	<b>94.34<sup>†‡</sup></b>	<b>89.57<sup>†</sup></b>	<b>90.53<sup>†</sup></b>	<b>92.83<sup>†</sup></b>	92.12	81.73	<b>89.72<sup>†</sup></b>	92.07	88.53	92.78	<b>90.19<sup>†</sup></b>	91.88	<b>85.88<sup>†‡</sup></b>	92.67	<b>90.35<sup>†</sup></b>
<b>+BERT</b>															
Single1O	95.20	91.64 <sup>†</sup>	90.87	93.55 <sup>†</sup>	92.01	81.95 <sup>†</sup>	90.44 <sup>†</sup>	92.56 <sup>†</sup>	<b>89.35</b>	93.44 <sup>†</sup>	90.89	91.78	86.13 <sup>†</sup>	92.51	90.88 <sup>†</sup>
Local1O	95.32	91.30	91.03	93.17	91.93	81.66	90.09	92.32	89.26	93.05	90.93	91.62	85.67	92.51	90.70
Single2O	95.31	<b>91.69<sup>†‡</sup></b>	<b>91.30<sup>†</sup></b>	<b>93.60<sup>†‡</sup></b>	<b>92.09<sup>†</sup></b>	<b>82.00<sup>†‡</sup></b>	<b>90.75<sup>†‡</sup></b>	<b>92.62<sup>†‡</sup></b>	89.32	<b>93.66<sup>†</sup></b>	<b>91.21</b>	<b>91.74</b>	<b>86.40<sup>†</sup></b>	92.61	<b>91.02<sup>†‡</sup></b>
Local2O	<b>95.34</b>	91.38	91.13	93.34 <sup>†</sup>	92.07 <sup>†</sup>	81.67	90.43 <sup>†</sup>	92.45 <sup>†</sup>	89.26	93.50 <sup>†</sup>	90.99	91.66	86.09 <sup>†</sup>	<b>92.66</b>	90.86 <sup>†</sup>

Table 3: LAS and standard deviations on test sets. We report results averaged over 5 runs. We use ISO 639-1 codes to represent languages from UD. <sup>†</sup> means that the model is statistically significantly better than the **Local1O** model by Wilcoxon rank-sum test with a significance level of  $p < 0.05$ . We use <sup>‡</sup> to represent winner of the significant test between the **Single2O** and **Local2O** models.

System	Train	Test	Time Complexity
GNN	392	464	$O(n^2d)$
Zhang et al. (2020)	200	400	$O(n^3)$
Single1O	616	1123	$O(n^2)$
Local1O	<b>625</b>	<b>1150</b>	$O(n^2)$
Single2O	481	966	$O(n^3)$
Local2O	486	1006	$O(n^3)$

Table 4: Comparison of training and testing speed (sentences per second) and the time complexity of the decoders of different approaches on PTB.

### 3.4 Speed Comparison

We evaluate the speed of different approaches on a single GeForce GTX 1080 Ti GPU following the setting of Zhang et al. (2020). As shown in Table 4, our **Local** approach and **Single** approach have almost the same speed. Our second-order approaches only slow down the training and testing speed in comparison with the first-order approaches by 23% and 12% respectively. They are also significantly faster than previous state-of-the-art approaches. Our **Local** approach is 1.2 and 2.3 times faster than **GNN** in training and testing respectively and is 2.4 and 2.9 times faster than the second-order tree CRF approach of Zhang et al. (2020).

In terms of time complexity, our second-order decoders have a time complexity of  $O(n^3)$ <sup>7</sup>; while the time complexity of **GNN** is  $O(n^2d)$ , the hidden size  $d$  (500 by default) is typically much larger than sentence length  $n$ ; and the decoder of Zhang et al. (2020) has a time complexity of  $O(n^3)$  as well, but it requires sequential computation over the input sentence while our decoders can be parallelized

<sup>7</sup>The MST algorithm has a time complexity of  $O(n^2)$  and we follow Dozat et al. (2017) only using the MST algorithm when the argmax predictions of structured output are not trees.

over words of the input sentence.

## 4 Conclusion

We propose second-order graph-based dependency parsing based on message passing and end-to-end neural networks. We modify a previous approach that predicts dependency edges independently and also design a new approach that incorporates the head-selection structured constraint. Our experiments show that our second-order approaches have better overall performance than the first-order baselines; they achieve competitive accuracy with very recent start-of-the-art second-order graph-based parsers and are significantly faster. Our empirical comparisons also show that second-order decoders still outperform first-order decoders even with BERT embeddings, and that the usefulness of the head-selection constraint is limited, especially when using BERT embeddings. Our code is publicly available at [https://github.com/wangxinyu0922/Second\\_Order\\_Parsing](https://github.com/wangxinyu0922/Second_Order_Parsing).

## Acknowledgements

This work was supported by the National Natural Science Foundation of China (61976139).

## References

- Joakim Nivre et al. 2018. Universal dependencies 2.2. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.
- Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

- Xavier Carreras. 2007. Experiments with a higher-order projective dependency parser. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*.
- Hao Cheng, Hao Fang, Xiaodong He, Jianfeng Gao, and Li Deng. 2016. Bi-directional attention with agreement for dependency parsing. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2204–2214, Austin, Texas. Association for Computational Linguistics.
- Kevin Clark, Minh-Thang Luong, Christopher D. Manning, and Quoc Le. 2018. Semi-supervised sequence modeling with cross-view training. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1914–1925, Brussels, Belgium. Association for Computational Linguistics.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- Timothy Dozat and Christopher D Manning. 2017. Deep biaffine attention for neural dependency parsing. In *International Conference on Learning Representations*.
- Timothy Dozat and Christopher D. Manning. 2018. Simpler but more accurate semantic dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 484–490, Melbourne, Australia. Association for Computational Linguistics.
- Timothy Dozat, Peng Qi, and Christopher D. Manning. 2017. Stanford’s graph-based neural dependency parser at the CoNLL 2017 shared task. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 20–30, Vancouver, Canada. Association for Computational Linguistics.
- Daniel Fernández-González and Carlos Gómez-Rodríguez. 2019. Left-to-right dependency parsing with pointer networks. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 710–716, Minneapolis, Minnesota. Association for Computational Linguistics.
- Erick Fonseca and André F. T. Martins. 2020. Revisiting higher-order dependency parsers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.
- Matthew R. Gormley, Mark Dredze, and Jason Eisner. 2015. Approximation-aware dependency parsing by belief propagation. *Transactions of the Association for Computational Linguistics*, 3:489–501.
- Tao Ji, Yuanbin Wu, and Man Lan. 2019. Graph-based dependency parsing with graph neural networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2475–2485, Florence, Italy. Association for Computational Linguistics.
- Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.
- Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional LSTM feature representations. *Transactions of the Association for Computational Linguistics*, 4:313–327.
- Terry Koo and Michael Collins. 2010. Efficient third-order dependency parsers. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1–11, Uppsala, Sweden. Association for Computational Linguistics.
- Adhiguna Kuncoro, Miguel Ballesteros, Lingpeng Kong, Chris Dyer, and Noah A. Smith. 2016. Distilling an ensemble of greedy dependency parsers into one MST parser. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1744–1753, Austin, Texas. Association for Computational Linguistics.
- Wang Ling, Chris Dyer, Alan W. Black, and Isabel Trancoso. 2015. Two/too simple adaptations of Word2Vec for syntax problems. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1299–1304, Denver, Colorado. Association for Computational Linguistics.
- Xuezhe Ma and Eduard Hovy. 2017. Neural probabilistic model for non-projective MST parsing. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 59–69, Taipei, Taiwan. Asian Federation of Natural Language Processing.
- Xuezhe Ma, Zecong Hu, Jingzhou Liu, Nanyun Peng, Graham Neubig, and Eduard Hovy. 2018. Stack-pointer networks for dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1403–1414, Melbourne, Australia. Association for Computational Linguistics.
- Xuezhe Ma and Hai Zhao. 2012. Fourth-order dependency parsing. In *Proceedings of COLING 2012: Posters*, pages 785–796, Mumbai, India. The COLING 2012 Organizing Committee.

- Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. **Building a large annotated corpus of English: The Penn Treebank.** *Computational Linguistics*, 19(2):313–330.
- André Martins, Miguel Almeida, and Noah A. Smith. 2013. **Turning on the turbo: Fast third-order non-projective turbo parsers.** In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 617–622, Sofia, Bulgaria. Association for Computational Linguistics.
- André Martins, Noah Smith, Mário Figueiredo, and Pedro Aguiar. 2011. **Dual decomposition with many overlapping components.** In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 238–249, Edinburgh, Scotland, UK. Association for Computational Linguistics.
- Ryan McDonald and Fernando Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *11th Conference of the European Chapter of the Association for Computational Linguistics*.
- Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005. **Non-projective dependency parsing using spanning tree algorithms.** In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 523–530, Vancouver, British Columbia, Canada. Association for Computational Linguistics.
- Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.
- Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. **Deep contextualized word representations.** In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.
- Peng Qi, Timothy Dozat, Yuhao Zhang, and Christopher D. Manning. 2018. **Universal dependency parsing from scratch.** In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 160–170, Brussels, Belgium. Association for Computational Linguistics.
- Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. 2018. **On the convergence of adam and beyond.** In *International Conference on Learning Representations*.
- David Smith and Jason Eisner. 2008. **Dependency parsing by belief propagation.** In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 145–156, Honolulu, Hawaii. Association for Computational Linguistics.
- Wenhui Wang and Baobao Chang. 2016. **Graph-based dependency parsing with bidirectional LSTM.** In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2306–2315, Berlin, Germany. Association for Computational Linguistics.
- Xinyu Wang, Jingxian Huang, and Kewei Tu. 2019. **Second-order semantic dependency parsing with end-to-end neural networks.** In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4609–4618, Florence, Italy. Association for Computational Linguistics.
- Nianwen Xue, Fu-Dong Chiou, and Martha Palmer. 2002. **Building a large-scale annotated Chinese corpus.** In *COLING 2002: The 19th International Conference on Computational Linguistics*.
- Yu Zhang, Zhenghua Li, and Min Zhang. 2020. Efficient second-order treecrf for neural dependency parsing. *arXiv preprint arXiv:2005.00975*.
- Zhisong Zhang, Xuezhe Ma, and Eduard Hovy. 2019. **An empirical investigation of structured output modeling for graph-based neural dependency parsing.** In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5592–5598, Florence, Italy. Association for Computational Linguistics.
- Junru Zhou and Hai Zhao. 2019. **Head-driven phrase structure grammar parsing on Penn treebank.** In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2396–2408, Florence, Italy. Association for Computational Linguistics.

# High-order Refining for End-to-end Chinese Semantic Role Labeling

Hao Fei<sup>1</sup>, Yafeng Ren<sup>2</sup> and Donghong Ji<sup>1\*</sup>

1. Department of Key Laboratory of Aerospace Information Security and Trusted Computing,  
Ministry of Education, School of Cyber Science and Engineering, Wuhan University, China

2. Guangdong University of Foreign Studies, China

{hao.fei, renyafeng, dhji}@whu.edu.cn

## Abstract

Current end-to-end semantic role labeling is mostly accomplished via graph-based neural models. However, these all are first-order models, where each decision for detecting any predicate-argument pair is made in isolation with local features. In this paper, we present a high-order refining mechanism to perform interaction between all predicate-argument pairs. Based on the baseline graph model, our high-order refining module learns higher-order features between all candidate pairs via attention calculation, which are later used to update the original token representations. After several iterations of refinement, the underlying token representations can be enriched with globally interacted features. Our high-order model achieves state-of-the-art results on Chinese SRL data, including CoNLL09 and Universal Proposition Bank, meanwhile relieving the long-range dependency issues.

## 1 Introduction

Semantic role labeling (SRL), as the shallow semantic parsing aiming to detect the semantic predicates and their argument roles in texts, plays a core role in natural language processing (NLP) community (Pradhan et al., 2005; Zhao et al., 2009; Lei et al., 2015; Xia et al., 2019b). SRL is traditionally handled by two pipeline steps: predicate identification (Scheible, 2010) and argument role labeling (Pradhan et al., 2005). More recently, growing interests are paid for developing end-to-end SRL, achieving both two subtasks, i.e., recognizing all possible predicates together with their arguments jointly, via one single model (He et al., 2018a).

The end-to-end joint architecture can greatly alleviate the error propagation problem, thus helping to achieve better task performance. Currently, the end-to-end SRL methods largely are graph-based

neural models, enumerating all possible predicates and their arguments exhaustively (He et al., 2018a; Cai et al., 2018; Li et al., 2019). However, these first-order models that only consider one predicate-argument pair at a time can be limited to short-term features and local decisions, thus being subjective to long-range dependency issues existing at large surface distances between arguments (Chen et al., 2019; Lyu et al., 2019). This makes it imperative to capture the global interactions between multiple predicates and arguments.

In this paper, based on the graph-based model architecture, we propose to further learn the higher-order interaction between all predicate-argument pairs by performing iterative refining for the underlying token representations. Figure 1 illustrates the overall framework of our method. The BiLSTM encoder (Hochreiter and Schmidhuber, 1997) first encodes the inputs into the initial token representations for producing predicate and argument representations, respectively. The biaffine attention then exhaustively calculates the score representations for all the candidate predicate-argument pairs. Based on all these score representations, our high-order refining module generates high-order feature for each corresponding token via an attention mechanism, which is then used for upgrading the raw token representation. After total  $N$  iterations of the above refining procedure, the information between the predicates and the associated arguments can be fully interacted, and thus results in global consistency for SRL.

On the other hand, most of the existing SRL studies focus on the English language, while there is little work in Chinese, mainly due to the limited amount of annotated data. In this study, we focus on the Chinese SRL. We show that our proposed high-order refining mechanism can be especially beneficial for such lower-resource language. Meanwhile, our proposed refining process is fully

\*Corresponding author.

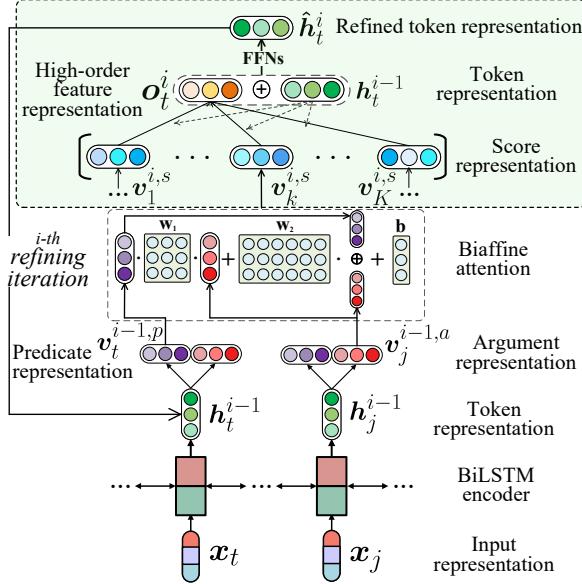


Figure 1: The overview of the graph-based high-order model for end-to-end SRL. The dotted-line green box is our proposed high-order refining module.

parallel and differentiable.

We conduct experiments on the dependency-based Chinese SRL datasets, including CoNLL09 (Hajič et al., 2009), and Universal Proposition Bank (Akbik et al., 2015; Akbik and Li, 2016). Results show that the graph-based end-to-end model with our proposed high-order refining consistently brings task improvements, compared with baselines, achieving state-of-the-art results for Chinese end-to-end SRL.

## 2 Related Work

Gildea and Jurafsky (2000) pioneer the task of semantic role labeling, as a shallow semantic parsing. Earlier efforts are paid for designing hand-crafted discrete features with machine learning classifiers (Pradhan et al., 2005; Punyakanok et al., 2008; Zhao et al., 2009). Later, a great deal of work takes advantages of neural networks with distributed features (FitzGerald et al., 2015; Roth and Lapata, 2016; Marcheggiani and Titov, 2017; Strubell et al., 2018). On the other hand, many previous work shows that integrating syntactic tree structure can greatly facilitate SRL (Marcheggiani et al., 2017; He et al., 2018b; Zhang et al., 2019; Fei et al., 2020b).

Prior studies traditionally separate SRL into two individual subtasks, i.e., predicate disambiguation and argument role labeling, mostly conducting only the argument role labeling based on the pre-

identified predicate (Pradhan et al., 2005; Zhao et al., 2009; FitzGerald et al., 2015; He et al., 2018b; Fei et al., 2020a). More recently, several researches consider the end-to-end solution that handles both two subtasks by one single model. All of them employs graph-based neural model, exhaustively enumerating all the possible predicate and argument mentions, as well as their relations (He et al., 2018a; Cai et al., 2018; Li et al., 2019; Xia et al., 2019a). Most of these end-to-end models, however, are first-order, considering merely one predicate-argument pair at a time. In this work, we propose a high-order refining mechanism to reinforce the graph-based end-to-end method.

Note that most of the existing SRL work focuses on the English language, with less for Chinese, mainly due to the limited amount of annotated data (Xia et al., 2019a). In this paper, we aim to improve the Chinese SRL and make compensation of the data scarcity by our proposed high-order model.

## 3 Framework

**Task formulation.** Following prior end-to-end SRL work (He et al., 2018a; Li et al., 2019), we treat the task as predicate-argument-role triplets prediction. Given an input sentence  $S = \{w_1, \dots, w_n\}$ , the system is expected to output a set of triplets  $\mathcal{Y} \in \mathcal{P} \times \mathcal{A} \times \mathcal{R}$ , where  $\mathcal{P} = \{p_1, \dots, p_m\}$  are all possible predicate tokens,  $\mathcal{A} = \{a_1, \dots, a_l\}$  are all associated argument tokens, and  $\mathcal{R}$  are the corresponding role labels for each  $a_i$ , including a null label  $\epsilon$  indicating no relation between a pair of predicate argument.

### 3.1 Baseline Graph-based SRL Model

Our baseline SRL model is mostly from He et al. (2018a). First, we obtain the vector representation  $x_t^w$  of each word  $w_t$  from pre-trained embeddings. We then make use of the part-of-speech (POS) tag for each word, and use its embedding  $x_t^{pos}$ . A convolutional neural networks (CNNs) is used to encode Chinese characters inside a word  $x_t^c$ . We concatenate them as input representations:  $x_t = [x_t^w; x_t^{pos}; x_t^c]$ .

Thereafter, a multi-layer bidirectional LSTM (BiLSTM) is used to encode the input representations into contextualized token representations:  $h_1, \dots, h_n = \text{BiLSTM}(x_1, \dots, x_n)$ . Based on the token representations, we further generate the separate predicate representations and argument representations:  $v_t^p = \text{FFN}(h_t)$ ,  $v_t^a = \text{FFN}(h_t)$ .

Then, a biaffine attention (Dozat and Manning, 2016) is used for scoring the semantic relationships exhaustively over all the predicate-argument pairs:

$$\mathbf{v}^s(p_i, a_j) = \mathbf{v}_i^p \cdot \mathbf{W}_1 \cdot \mathbf{v}_j^a + \mathbf{W}_2 \cdot [\mathbf{v}_i^p; \mathbf{v}_j^a] + \mathbf{b}, \quad (1)$$

where  $\mathbf{W}_1$ ,  $\mathbf{W}_2$  and  $\mathbf{b}$  are parameters.

**Decoding and learning.** Once a predicate-argument pair  $(p_i, a_j)$  (i.e., the role label  $r \neq \epsilon$ ) is determined by a softmax classifier, based on the score representation  $\mathbf{v}^s(p_i, a_j)$ , the model outputs this tuple  $(p, a, r)$ .

During training, we optimize the probability  $P_\theta(\hat{y}|S)$  of the tuple  $y_{(p_i, a_j, r)}$  over a sentence  $S$ :

$$\begin{aligned} P_\theta(y|S) &= \prod_{p \in \mathcal{P}, a \in \mathcal{A}, r \in \mathcal{R}} P_\theta(y_{(p, a, r)}|S) \\ &= \prod_{p \in \mathcal{P}, a \in \mathcal{A}, r \in \mathcal{R}} \frac{\phi(p, a, r)}{\sum_{\hat{r} \in \mathcal{R}} \phi(p, a, \hat{r})}, \end{aligned} \quad (2)$$

where  $\theta$  is the parameters of the model and  $\phi(p, a, r)$  represents the total unary score from:

$$\begin{aligned} \phi(p, a, r) &= \mathbf{W}_p \text{ReLU}(\mathbf{v}^p) + \mathbf{W}_a \text{ReLU}(\mathbf{v}^a) \\ &\quad + \mathbf{W}_s \text{ReLU}(\mathbf{v}^s(p, a)). \end{aligned} \quad (3)$$

The final objective is to minimize the negative log-likelihood of the golden structure:

$$\mathcal{L} = -\log P_\theta(y|S). \quad (4)$$

### 3.2 Higher-order Refining

The baseline graph model is a first-order model, since it only considers one predicate-argument pair (as in Eq. 3) at a time. This makes it limited to short-term and local decisions, and thus subjective to long-distance dependency problem whenever there are larger surface distances between arguments. We here propose a higher-order refining mechanism for allowing a deep interactions between all predicate-argument pairs.

Our high-order model is shown in Figure 1. Compared with the baseline model, the main difference lies in the high-order refining module. Our motivation is to inform each predicate-argument pair with the information of the other rest of pairs from the global viewpoint. We reach this by refining the underlying token representations  $\mathbf{h}_t$  with refined ones which carry high-order interacted features.

Concretely, we take the baseline as the initiation, performing refinement iteratively. At the  $i$ -th refining iteration, we can collect the score representations  $\mathbf{V}^{i,s} = \{\mathbf{v}_1^{i,s}, \dots, \mathbf{v}_K^{i,s}\}$  of all candidate

predicate-argument pairs, where  $K$  (i.e.,  $\binom{n}{2}$ ) are the total combination number of these pairs. Based on  $\mathbf{V}^{i,s}$ , we then generate the high-order feature vector  $\mathbf{o}_t^i$  by using an attention mechanism guided by the current token representation  $\mathbf{h}_t^{i-1}$  for word  $w_t$  at last turn, i.e., the  $(i-1)$ -th iteration:

$$\begin{aligned} u_k^i &= \tanh(\mathbf{W}_3 \mathbf{h}_t^{i-1} + \mathbf{W}_4 \mathbf{v}_k^{i,s}), \\ \alpha_k^i &= \text{softmax}(u_k^i), \\ \mathbf{o}_t^i &= \sum_{k=1}^K \alpha_k^i \mathbf{v}_k^{i,s}, \end{aligned} \quad (5)$$

where  $\mathbf{W}_3$  and  $\mathbf{W}_4$  are parameters. We then concatenate the raw token representation and high-order feature representation together, and obtain the refined token representation after a non-linear projection  $\hat{\mathbf{h}}_t^i = \text{FFN}([\mathbf{o}_t^i; \mathbf{h}_t^{i-1}])$ . Finally, we use  $\hat{\mathbf{h}}_t^i$  to update the old one  $\mathbf{h}_t^{i-1}$ . After total  $N$  iterations of high-order refinement, we expect the model to capture more informative features at global scope and achieve the global consistency.

## 4 Experiments

### 4.1 Settings

Our method is evaluated on the Chinese SRL benchmarks, including CoNLL09<sup>1</sup> and Universal Proposition Bank (UPB)<sup>2</sup>. Each dataset comes with its own training, development and test sets. Precision, recall and F1 score are used as the metrics.

We use the pre-trained Chinese fasttext embeddings<sup>3</sup>. The BiLSTM has hidden size of 350, with three layers. The kernel sizes of CNN are [3,4,5]. We adopt the Adam optimizer with initial learning rate of 1e-5. We train the model by mini-batch size in [16,32] with early-stop strategy. We also use the contextualized Chinese word representations, i.e., ELMo<sup>4</sup> and BERT (Chinese-base-version)<sup>5</sup>.

### 4.2 Main Results

We mainly make comparisons with the recent end-to-end SRL models, as well as the pipeline methods on standalone argument role labeling given the gold predicates. Table 1 shows the results on the Chinese CoNLL09. We first find that the joint detection for predicates and arguments can be more beneficial

<sup>1</sup><https://catalog.ldc.upenn.edu/LDC2012T03>

<sup>2</sup><https://github.com/System-T/UniversalPropositions>

<sup>3</sup><https://fasttext.cc/>

<sup>4</sup><https://github.com/HIT-SCIR/ELMoForManyLangs>

<sup>5</sup><https://github.com/google-research/bert>

	Arg.			Prd.
	P	R	F1	F1
<b>• Pipeline method</b>				
Zhao et al. (2009)	80.4	75.2	77.7	-
Björkelund et al. (2009)	82.4	75.1	78.6	-
Roth and Lapata (2016)	83.2	75.9	79.4	-
Marcheggiani and Titov (2017)	84.6	80.4	82.5	-
He et al. (2018b)	84.2	81.5	82.8	-
Cai and Lapata (2019) <sup>‡</sup>	85.4	84.6	85.0	-
<b>• End-to-end method</b>				
He et al. (2018a)	82.6	83.6	83.0	85.7
Cai et al. (2018)	84.7	84.0	84.3	86.0
Li et al. (2019)	84.9	84.6	84.8	86.9
Xia et al. (2019a)	84.6	85.7	85.1	87.2
+BERT	88.0	89.1	88.5	89.6
Ours	<b>85.7</b>	<b>86.2</b>	<b>85.9</b>	<b>88.6</b>
+ELMo	86.4	87.6	87.1	88.9
+BERT	<b>87.4</b>	<b>89.3</b>	<b>88.8</b>	<b>90.3</b>

Table 1: Performances on CoNLL09. Results with <sup>‡</sup> indicates the additional resources are used.

	P	R	F1
He et al. (2018a)	64.8	65.3	64.9
Cai et al. (2018)	65.0	66.4	65.8
Li et al. (2019)	65.4	67.2	66.0
Xia et al. (2019a)	65.2	67.6	66.1
Ours	<u>67.5</u>	<u>68.8</u>	<u>67.9</u>
+ELMo	68.0	70.6	68.8
+BERT	<b>70.0</b>	<b>73.0</b>	<b>72.4</b>

Table 2: Performances by end-to-end models for the argument role labeling on UPB.

than the pipeline detection of SRL, notably with 85.1% F1 score on argument detection by Xia et al. (2019a). Most importantly, our high-order end-to-end model outperforms all these baselines on both two subtasks, with 85.9% F1 score for argument role labeling and 88.6% F1 score for predicate detection. When the contextualized word embeddings are available, we find that our model can achieve further improvements, i.e., 88.8% and 90.3% F1 scores for two subtasks, respectively.

Table 2 shows the performances on UPB. Overall, the similar trends are kept as that on CoNLL09. Our high-order model still performs the best, yielding 67.9% F1 score on argument role labeling, verifying its prominent capability for the SRL task. Also with BERT embeddings, our model further wins a great advance of performances.

### 4.3 Analysis

**High-order refinement.** We take a further step, looking into our proposed high-order refining

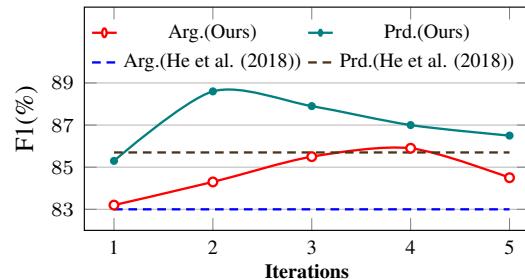


Figure 2: Performances by varying refining iterations.

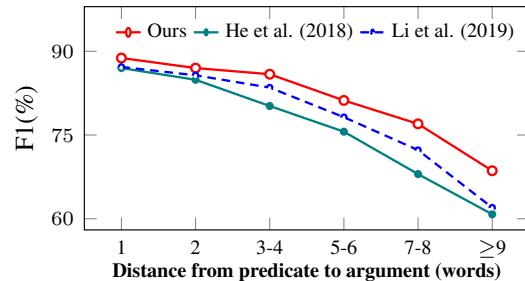


Figure 3: Argument recognition under varying surface distance between predicates and arguments.

mechanism. We examine the performances under varying refining iterations in Figure 2. Compared with the first-order baseline model by He et al. (2018a), our high-order model can achieve better performances for both two subtasks. We find that our model can reach the peak for predicate detection with total 2 iterations of refinement, while the best iteration number is 4 for argument labeling.

**Long-distance dependencies.** Figure 3 shows the performances of argument recognition by different surface distance between predicates and arguments. The overall results decrease when arguments are farther away from the predicates. Nevertheless, our high-order model can beat against such drop significantly. Especially when the distance grows larger, e.g., distance  $\geq 7$ , the winning score by our model even becomes more notable.

## 5 Conclusion

We proposed a high-order end-to-end model for Chinese SRL. Based on the baseline graph-based model, our high-order refining module performed interactive learning between all predicate-argument pairs via attention calculation. The generated higher-order featured token representations then were used to update the original ones. After total  $N$  iterations of refinement, we enriched the underlying token representations with global interactions, and made the learnt features more informative. Our

high-order model brought state-of-the-art results on Chinese SRL data, i.e., CoNLL09 and Universal Proposition Bank, meanwhile relieving the long-range dependency issues.

## Acknowledgments

We thank the anonymous reviewers for their valuable and detailed comments. This work is supported by the National Natural Science Foundation of China (No. 61772378, No. 61702121), the National Key Research and Development Program of China (No. 2017YFC1200500), the Research Foundation of Ministry of Education of China (No. 18JZD015), the Major Projects of the National Social Science Foundation of China (No. 11&ZD189), the Key Project of State Language Commission of China (No. ZDI135-112) and Guangdong Basic and Applied Basic Research Foundation of China (No. 2020A151501705).

## References

- Alan Akbik, Laura Chiticariu, Marina Danilevsky, Yunyao Li, Shivakumar Vaithyanathan, and Huaiyu Zhu. 2015. Generating high quality proposition banks for multilingual semantic role labeling. In *Proceedings of the ACL*, pages 397–407.
- Alan Akbik and Yunyao Li. 2016. Polyglot: Multilingual semantic role labeling with unified labels. In *Proceedings of the ACL*, pages 1–6.
- Anders Björkelund, Love Hafnell, and Pierre Nugues. 2009. Multilingual semantic role labeling. In *Proceedings of the CoNLL*, pages 43–48.
- Jiaxun Cai, Shexia He, Zuchao Li, and Hai Zhao. 2018. A full end-to-end semantic role labeler, syntactic-agnostic over syntactic-aware? In *Proceedings of the CoLing*, pages 2753–2765.
- Rui Cai and Mirella Lapata. 2019. Semi-supervised semantic role labeling with cross-view training. In *Proceedings of the EMNLP*, pages 1018–1027.
- Xinchi Chen, Chunchuan Lyu, and Ivan Titov. 2019. Capturing argument interaction in semantic role labeling with capsule networks. In *Proceedings of the EMNLP*, pages 5415–5425.
- Timothy Dozat and Christopher D Manning. 2016. Deep biaffine attention for neural dependency parsing. *arXiv preprint arXiv:1611.01734*.
- Hao Fei, Meishan Zhang, and Donghong Ji. 2020a. Cross-lingual semantic role labeling with high-quality translated training corpus. In *Proceedings of the ACL*, pages 7014–7026.
- Hao Fei, Meishan Zhang, Fei Li, and Donghong Ji. 2020b. Cross-lingual semantic role labeling with model transfer. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2427–2437.
- Nicholas FitzGerald, Oscar Täckström, Kuzman Ganchev, and Dipanjan Das. 2015. Semantic role labeling with neural network factors. In *Proceedings of the EMNLP*, pages 960–970.
- Daniel Gildea and Daniel Jurafsky. 2000. Automatic labeling of semantic roles. In *Proceedings of the ACL*, pages 512–520.
- Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Stráňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the CoNLL*, pages 1–18.
- Luheng He, Kenton Lee, Omer Levy, and Luke Zettlemoyer. 2018a. Jointly predicting predicates and arguments in neural semantic role labeling. In *Proceedings of the ACL*, pages 364–369.
- Shexia He, Zuchao Li, Hai Zhao, and Hongxiao Bai. 2018b. Syntax for semantic role labeling, to be, or not to be. In *Proceedings of the ACL*, pages 2061–2071.
- Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.
- Tao Lei, Yuan Zhang, Lluís Màrquez, Alessandro Moschitti, and Regina Barzilay. 2015. High-order low-rank tensors for semantic role labeling. In *Proceedings of the NAACL*, pages 1150–1160.
- Zuchao Li, Shexia He, Hai Zhao, Yiqing Zhang, Zhuosheng Zhang, Xi Zhou, and Xiang Zhou. 2019. Dependency or span, end-to-end uniform semantic role labeling. In *Proceedings of the AAAI*, pages 6730–6737.
- Chunchuan Lyu, Shay B. Cohen, and Ivan Titov. 2019. Semantic role labeling with iterative structure refinement. In *Proceedings of EMNLP*, pages 1071–1082.
- Diego Marcheggiani, Anton Frolov, and Ivan Titov. 2017. A simple and accurate syntax-agnostic neural model for dependency-based semantic role labeling. In *Proceedings of the CoNLL*, pages 411–420.
- Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of the EMNLP*, pages 1506–1515.
- Sameer Pradhan, Wayne Ward, Kadri Hacioglu, James Martin, and Daniel Jurafsky. 2005. Semantic role labeling using different syntactic views. In *Proceedings of the ACL*, pages 581–588.

Vasin Punyakanok, Dan Roth, and Wen-tau Yih. 2008. The importance of syntactic parsing and inference in semantic role labeling. *Computational Linguistics*, 34(2):257–287.

Michael Roth and Mirella Lapata. 2016. Neural semantic role labeling with dependency path embeddings. In *Proceedings of the ACL*, pages 1192–1202.

Christian Scheible. 2010. An evaluation of predicate argument clustering using pseudo-disambiguation. In *Proceedings of the LREC*, pages 1187–1194.

Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. 2018. Linguistically-informed self-attention for semantic role labeling. In *Proceedings of the EMNLP*, pages 5027–5038.

Qingrong Xia, Zhenghua Li, and Min Zhang. 2019a. A syntax-aware multi-task learning framework for Chinese semantic role labeling. In *Proceedings of the EMNLP*, pages 5382–5392.

Qingrong Xia, Zhenghua Li, Min Zhang, Meishan Zhang, Guohong Fu, Rui Wang, and Luo Si. 2019b. Syntax-aware neural semantic role labeling. In *Proceedings of the AAAI*, pages 7305–7313.

Yue Zhang, Rui Wang, and Luo Si. 2019. Syntax-enhanced self-attention-based semantic role labeling. In *Proceedings of the EMNLP*, pages 616–626.

Hai Zhao, Wenliang Chen, and Chunyu Kit. 2009. Semantic dependency parsing of NomBank and PropBank: An efficient integrated approach via a large-scale feature selection. In *Proceedings of the EMNLP*, pages 30–39.

# Exploiting WordNet Synset and Hypernym Representations for Answer Selection

Weikang Li and Yunfang Wu<sup>\*</sup>

MOE Key Lab of Computational Linguistics, School of EECS, Peking University

{wavejkd, wuyf}@pku.edu.cn

## Abstract

Answer selection (AS) is an important subtask of document-based question answering (DQA). In this task, the candidate answers come from the same document, and each answer sentence is semantically related to the given question, which makes it more challenging to select the true answer. WordNet provides powerful knowledge about concepts and their semantic relations, so we employ WordNet to enrich the abilities of paraphrasing and reasoning of the network-based question answering model. Specifically, we exploit the synset and hypernym concepts to enrich the word representation and incorporate the similarity scores of two concepts that share the synset or hypernym relations into the attention mechanism. The proposed WordNet-enhanced hierarchical model (WEHM) consists of four modules, including WordNet-enhanced word representation, sentence encoding, WordNet-enhanced attention mechanism, and hierarchical document encoding. Extensive experiments on the public WikiQA and SelQA datasets demonstrate that our proposed model significantly improves the baseline system and outperforms all existing state-of-the-art methods by a large margin.

## 1 Introduction

Answer selection (AS) is a challenging subtask of document-based question answering (DQA) in natural language processing (NLP). The AS task is to select a whole answer sentence from the document and can be regarded as a ranking problem, which is different from the machine reading comprehension (MRC) task on the SQuAD and MS-MARCO datasets. Compared with a single word or phrase, returning the full sentence often adds more value as the user can easily verify the correctness without reading a lengthy document (Yih et al., 2013). In

this paper, we focus on the AS task of DQA. Table 1 gives a real example of this task.

Lots of fruits on answer selection have been achieved via deep learning models, including convolutional neural network (CNN) (Yang et al., 2015), recurrent neural network (RNN) (Tan et al., 2015), attention-way (Wang et al., 2016) and generative adversarial networks (GAN) (Wang et al., 2017a). Recently proposed models often consist of an embedding layer, an encoding layer, an interaction layer, and an answer layer (Weissenborn et al., 2017; Wang et al., 2017b; Hewlett et al., 2017).

Different from other question answering like community-based question answering, the candidate answers of DQA come from the same document, and each candidate answer is semantically related to the question. From the example in Table 1, we can see that almost every candidate answer contains the information related to the word “food” and “afghan” in the given question. As a result, it is difficult for the existing network-based models to choose the right answer, since the power generation ability of the networks may have transformed the sentences into similar meanings in the latent space.

To tackle this challenge, we propose to leverage WordNet knowledge base into the neural network model. Our hypothesis is that the ability of paraphrase and reasoning is essential to the question-answering task. WordNet is a semantic network (Fellbaum, 1998), where the words that are related in meanings are interlinked by means of pointers, which stand for different semantic relations. It organizes concepts mainly with the is-a relation, where a concept is a set of word senses (synset). On the one hand, we apply the synset information to enrich the sentence’s paraphrase representation, which could distinguish the candidate answers in the latent semantic space to some degree. On the other hand, we apply the hypernym information to capture reasoning knowledge. The real case

\* Corresponding author.

<b>Question:</b> what food is in afghan ?
<b>Document:</b>
[1] A table setting of Afghan food in Kabul.
[2] Afghan cuisine is largely based upon the nation's chief crops; cereals like wheat, maize, barley and rice.
[3] .....
[4] Afghanistan's culinary specialties reflect its ethnic and geographic diversity.
[5] Though it has similarities with neighboring countries, Afghan cuisine is undeniably unique.
[6] .....
<b>Reference Answer:</b>
Afghan cuisine is largely based upon the nation's chief crops; cereals like wheat, maize, barley and rice.

Table 1: An example from the WikiQA data. The text is shown in its original form, which may contain errors in typing.

from the WikiQA dataset in table 1 shows that if our model has the ability of reasoning on common sense, like “wheat is a kind of food”, “maize is a kind of food” and so on, it would be of great help for choosing the right answer with respect to the question “what food is in afghan ?”.

The overall framework of our proposed model is shown in Figure 1, which mainly consists of four modules. First, we apply the synset and hypernym information to enrich the word representation. Second, we use an RNN module to encode the WordNet-enhanced word representation. Third, we propose to use the synset’s and hypernym’s relation score based on two senses’ path in the WordNet to enrich the attention mechanism. Specifically, the attention similarity matrix is not only measured by a similarity score over hidden vectors produced by CNN or RNN networks but also measured based on the synset and hypernym relation scores of two concepts in Wordnet. And then following the compare-aggregate framework (Wang and Jiang, 2016), we combine the original representation with the attention representation. Finally, considering the strong relations among context sentences, we employ a hierarchical neural network for answer sentence selection.

We conduct extensive experiments on the public WikiQA and SelQA datasets. The results show that our proposed WordNet-enhanced hierarchical model outperforms the baseline models by a large margin and achieves state-of-the-art performance on both datasets. On the WikiQA data, it obtains a MAP of 77.02, which beats the existing best result by 1.62 points; on the SelQA data, it achieves a MAP of 91.71, which outperforms the previous best result by 2.57 points.

## 2 Model Description

Given a question  $q$  and the sentences  $a_i, i = 1, 2, \dots, S$  in a document  $d$ , our model aims

to select the best sentence which could answer the question.

### 2.1 WordNet-Enhanced Word Representation

Firstly, we map each word into the vector space. Different from directly using word embedding or the concatenation of word embedding and sum of its character embeddings, we propose to exploit the word’s hypernym and synset in the WorNet to enrich the word representation. Suppose  $w_j$  is the  $j_{th}$  word in a sequence,  $k_{sj}$  and  $k_{hj}$  represent the hypernym and synset in the WordNet with respect to the word  $w_j$ . The WordNet-enhanced word embedding is computed as follows:

$$k_j = [w_j; k_{sj}; k_{hj}] \quad (1)$$

$$k_{sj} = \frac{1}{|S|} \sum_{i=1}^{|S|} w_i^{k_{sj}} \quad (2)$$

$$k_{hj} = \frac{1}{|H|} \sum_{i=1}^{|H|} w_i^{k_{hj}} \quad (3)$$

where  $w_i^{k_{sj}}$  and  $w_i^{k_{hj}}$  represent word embeddings in the synset and hypernym concepts respectively;  $|S|$  and  $|H|$  denote the number of concepts in the synset and hypernym respectively. And ; means the concatenation operation.

We use  $k_j^q$  and  $k_j^{a_i}$  to represent the  $j_{th}$  word’s WordNet-enhanced embedding of the question and the  $i_{th}$  candidate answer sentence respectively.

### 2.2 Sentence Encoding

We encode the question and each sentence in the document into latent vectors using a Bi-directional Gated Recurrent Unit (Bi-GRU) network. The formulas of a GRU (Cho et al., 2014) are as follows:

$$r_j = \sigma(W_r k_j + U_r h_{j-1} + b_r) \quad (4)$$

$$z_j = \sigma(W_z k_j + U_z h_{j-1} + b_z) \quad (5)$$

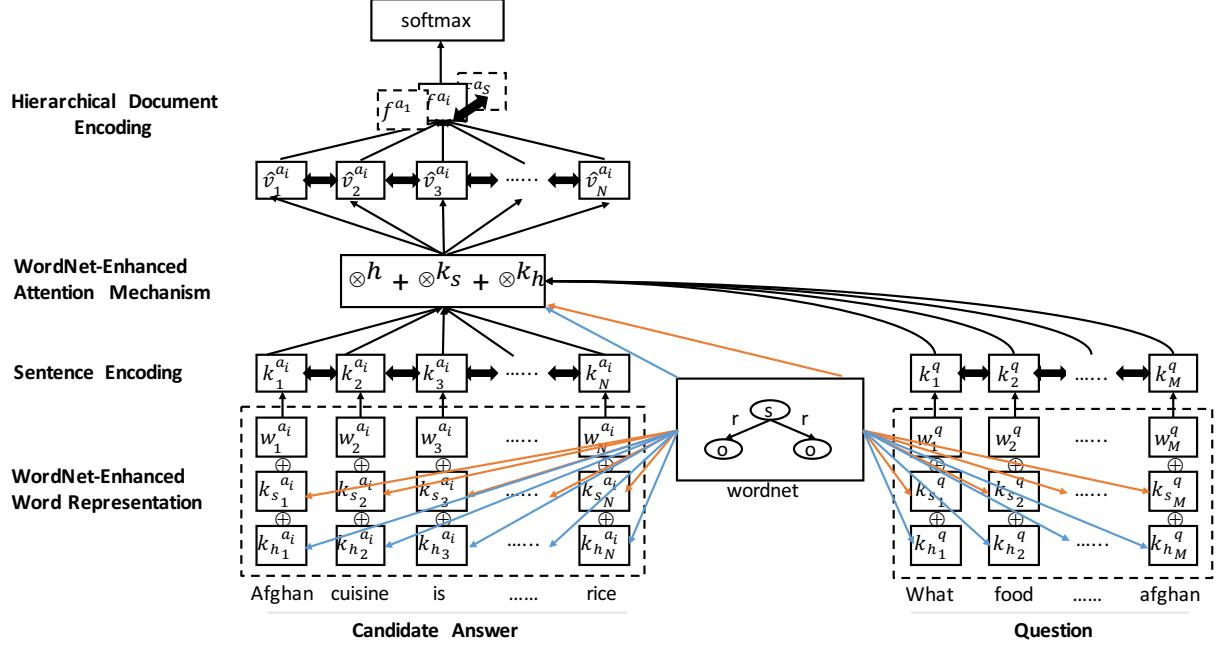


Figure 1: Framework of our proposed WordNet-enhanced hierarchical model (WEHM).

$$\tilde{h}_j = \tanh(W_h k_j + U_h(r_j \odot h_{j-1}) + b_h) \quad (6)$$

$$h_j = (1 - z_j) \odot h_{j-1} + z_j \odot \tilde{h}_j \quad (7)$$

where  $\odot$  is element-wise multiplication.  $r_j$  and  $z_j$  are the reset and update gates respectively. And  $W_r, W_z, W_h \in R^{H \times E}$ ,  $U_r, U_z, U_h \in R^{H \times H}$  and  $b_r, b_z, b_h \in R^{H \times 1}$  are parameters to be learned. A Bi-GRU processes the sequence in both forward and backward directions to produce two sequences  $[h_1^f, h_2^f, \dots, h_S^f]$  and  $[h_1^b, h_2^b, \dots, h_S^b]$ . The final output of  $h_j$  is the concatenation of  $h_j^f$  and  $h_j^b$ .

We use  $h_j^q$  and  $h_j^{a_i}$  to represent  $j_{th}$  word's hidden vector produced by sentence encoding in the question and in the  $i_{th}$  candidate answer sentence respectively.

### 2.3 WordNet-Enhanced Attention Mechanism

Different from the vanilla attention mechanism, where the attention score is only measured by hidden vectors, we propose to employ the synset and hypernym relation scores of two concepts in WordNet to enhance the attention mechanism, which can capture more rich interaction information between two sequences. The sketch of our proposed WordNet-enhanced attention mechanism is shown in Figure 2, which consists of three parts: the standard attention score, the synset relation score, and the hypernym relation score.

As for the standard attention mechanism, we adopt the Luong attention (also known as bilinear function attention mechanism) (Luong et al., 2015), which is widely used in NLP. In our model,  $M_{|a_i|, |q|}^h$  represents the attention score between the question and one of its candidate answers. The formulas of computing each element are as follows:

$$M_{n,m}^h = h_n^{a_i} W h_m^q {}^T \quad (8)$$

$$M_{n,m}^h = \exp(M_{n,m}^h) / \sum_{k=1}^{|q|} \exp(M_{n,k}^h) \quad (9)$$

where  $h_n^{a_i}$  and  $h_m^q$  represent the  $n_{th}$  and  $m_{th}$  word hidden vector in the candidate answer and the question respectively,  $|a_i|$  and  $|q|$  are the candidate answer's length and the question's length respectively.

Besides the standard attention, we employ two kinds of WordNet-enhanced mechanism to measure the attention score.

Lots of studies have been done on computing lexical similarity based on WordNet (Pedersen et al., 2004). Wu-Palmer Similarity (Wu and Palmer, 1994) denotes how similar two words senses are, based on the depth of the two senses in the taxonomy and that of their Least Common Subsumer. Leacock-Chodorow Similarity (Leacock and Chodorow, 1998) denotes how similar two-word senses are, based on the shortest path that connects the senses in the is-a (hypernym/hyponym) taxonomy.

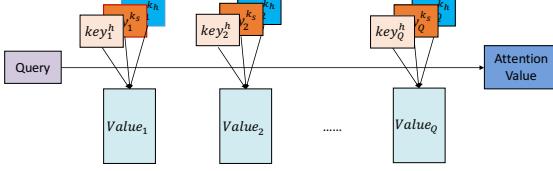


Figure 2: Sketch of our proposed WordNet-enhanced attention mechanism.  $Key_j^h$  means the attention score derived by two hidden vectors.  $Key_j^{k_s}$  and  $Key_j^{k_h}$  represent the attention score derived by synset relation and hypernym relation respectively.  $Value_j$  means the hidden vector of question, and  $Query$  means the candidate answer.

We use Wu-Palmer Similarity to compute the attention score with the synset relation.  $M_{|a_i||q|}^{k_s}$  represents the attention matrix between the question and one of its candidate answers, where each element  $M_{n,m}^{k_s}$  is computed as:

$$M_{n,m}^{k_s} = 2 * N_c / (N_{a_n^i} + N_{q_m} + 2 * N_c) \quad (10)$$

$$M_{n,m}^{k_s} = \exp(M_{n,m}^{k_s}) / \sum_{k=1}^{|q|} \exp(M_{n,k}^{k_s}) \quad (11)$$

where  $a_n^i$  and  $q_m$  represent the corresponding concepts of the  $n$ th word of the  $i$ th candidate answer and the  $m$ th word of the question respectively,  $c$  is the least common superconcept of  $a_n^i$  and  $q_m$ ,  $N_{a_n^i}$  is the number of nodes on the path from  $a_n^i$  to  $c$ ,  $N_{q_m}$  is the number of nodes on the path from  $N_{q_m}$  to  $c$ ,  $N_c$  is the number of nodes on the path from  $c$  to root.

We use Leacock-Chodorow Similarity to measure the attention score with hypernym relation. Let  $M_{|a_i||q|}^{k_h}$  denote the attention matrix between the question and one of its candidate answers, where each element  $M_{n,m}^{k_h}$  can be computed as:

$$M_{n,m}^{k_h} = -\log(path(a_n^i, q_m) / 2L) \quad (12)$$

$$M_{n,m}^{k_h} = \exp(M_{n,m}^{k_h}) / \sum_{k=1}^{|q|} \exp(M_{n,k}^{k_h}) \quad (13)$$

where  $path(a_n^i, q_m)$  is the shortest path length connecting two concepts and  $L$  is the whole taxonomy depth.

Finally, we combine all the three similarity matrixes. The formulas are as follows:

$$M_{n,m} = M_{n,m}^h + M_{n,m}^{k_s} + M_{n,m}^{k_h} \quad (14)$$

$$M_{n,m} = \exp(M_{n,m}) / \sum_{k=1}^{|q|} \exp(M_{n,k}) \quad (15)$$

Equipped with the WordNet-enhanced similarity matrix  $M$ , we apply the attention mechanism between the question encoding  $h^q$  and the sentence encoding  $h^{a_i}$  to obtain a new sentence representation  $v^{a_i}$ , which is a weighted sum of hidden vectors of the question. We then aggregate the vectors of  $h^{a_i}$  and  $v^{a_i}$ . Formulas are as follows:

$$v^{a_i} = M \cdot h^q \quad (16)$$

$$\hat{v}^{a_i} = [v^{a_i}; h^{a_i}; v^{a_i} \odot h^{a_i}; v^{a_i} + h^{a_i}; v^{a_i} - h^{a_i}] \quad (17)$$

where ; is the concatenation operation, + is element-wise addition, – is element-wise subtraction and  $\odot$  is element-wise multiplication.

## 2.4 Hierarchical Document Encoding

Inspired by the work (Bian et al., 2017), we also adopt a list-wise method to model the answer selection task. But different from their model, we employ a hierarchical Bi-GRU architecture to compare candidate sentences by ranking them with respect to a given question. Considering that candidate answers all come from a whole document, the hierarchical Bi-GRU architecture can capture contextual features among sentences and make the understanding of a document more coherent.

We first encode each candidate answer  $\hat{v}^{a_i}$  and then extract features among sentences' hidden vectors. Then we again encode the document based on each candidate answer's extracted features. The Bi-GRU is the same to that mentioned in our sentence encoding section.

$$u_j^{a_i} = BiGRU(u_{j-1}^{a_i}, \hat{v}_j^{a_i}) \quad (18)$$

$$u_{avg}^{a_i} = \frac{1}{|a_i|} \sum_{j=1}^{|a_i|} u_j^{a_i}, u_{max}^{a_i} = \max_{j=1}^{|a_i|} u_j^{a_i} \quad (19)$$

$$f^{a_i} = [u_{avg}^{a_i}; u_{max}^{a_i}] \quad (20)$$

$$\hat{u}_i^d = BiGRU(\hat{u}_{i-1}^d, f^{a_i}) \quad (21)$$

where  $j$  is the  $j$ th word in the  $i$ th sentence in the candidate answers,  $f^{a_i}$  is the  $i$ th sentence extracted features and  $\hat{u}_i^d$  is the  $i$ th sentence's hidden vector after the document encoding phase.

At last, we use a *softmax* layer to choose the right answer among every step's output of the document's RNN layer. The model is trained to minimize the cross-entropy loss function:

$$\tilde{a}_i = \sigma(FC(\hat{u}_i^d)) \quad (22)$$

Dataset	Split	#Questions	#Pairs
WikiQA	TRAIN	873	8672
	DEV	126	1130
	TEST	243	2351
SelQA	TRAIN	5529	66438
	DEV	785	9377
	TEST	1590	19435

Table 2: Statistical distribution of two benchmark datasets.

$$C = -\frac{1}{|d|} \sum_{i \in |d|} [a_i \log \tilde{a}_i + (1 - a_i) \log (1 - \tilde{a}_i)] \quad (23)$$

where  $FC$  is a feed-forward neural network,  $i$  means the sentence index in the document,  $|d|$  is the document’s length in terms of sentences,  $a_i$  is the true label (0 or 1) from the training data and  $\tilde{a}_i$  is the predicted probability score by our model. The sentence with the highest probability score is regarded as the right answer.

### 3 Experiments

#### 3.1 Datasets and Baselines

We use two different datasets to conduct our answer selection experiments: WikiQA (Yang et al., 2015) and SelQA (Jurczyk et al., 2016). Both datasets contain open-domain questions whose answers were extracted from Wikipedia articles. In the AS task, it is assumed that there is at least one correct answer for a question. In the WikiQA, there are some questions which have no answer, we removed these questions, just like other researches do. Table 2 shows the statistical distribution of the two datasets.

As for the WikiQA dataset, it has been well studied by lots of literature. Baselines adopted are as follows:

- **CNN-Cnt:** this model combines sentence representations produced by a convolutional neural network with the logistic regression (Yang et al., 2015).
- **ABCNN:** this model is an attention-based convolutional neural network (Yin et al., 2015).
- **IARNN-Occam:** this model adds regularization on the attention weights (Wang et al., 2016).

- **IARNN-Gate:** this model uses the question representation to build GRU gates for each candidate answer (Wang et al., 2016).
- **CubeCNN:** this model builds a CNN on all pairs of word similarities (He and Lin, 2016).
- **CA-Network:** this model applies a compare-aggregate neural network to model question answering problem (Wang and Jiang, 2016).
- **IWAN-Skip:** this model measures the similarity of sentence pairs by focusing on the interaction information (Shen et al., 2017b).
- **Dynamic-Clip:** this model proposes a novel attention mechanism named Dynamic-Clip Attention, which is then directly integrated into the Compare-Aggregate framework. (Bian et al., 2017).

As for the SelQA dataset, besides the above mentioned CNN-Cnt model, Jurczyk et al. (2016) also re-implement CNN-Tree and two attention RNN models. Other baselines are as follows:

- **CNN-hinge:** this is a re-implemented CNN-based model with hinge loss function (dos Santos et al., 2017).
- **CNN-DAN:** dos Santos et al. (2017) propose a CNN-based model trained with a DAN framework, which is to learn loss functions for predictors and also implements semi-supervised learning.
- **AdaQA:** Shen et al. (2017a) propose an adaptive question answering (AdaQA) model, which consists of a novel two-way feature abstraction mechanism to encapsulate co-dependent sentence representations.

The answer selection task can be considered as a ranking problem, and so two evaluation metrics are used: mean average precision (MAP) and mean reciprocal rank (MRR).

#### 3.2 Experiment Setup

The proposed models are implemented with TensorFlow. The dimension of word embeddings is set to 300. The word embeddings are initialized by 300D *GloVe 840B* (Pennington et al., 2014), and out-of-vocabulary words are initialized randomly. We fix the embeddings during training. We train the model with the Adam optimization algorithm with

Model	MAP	MRR
CNN-Cnt (Yang et al., 2015)	65.20	66.52
ABCNN (Yin et al., 2015)	69.21	71.08
CubeCNN (He and Lin, 2016)	70.90	72.34
IARNN-Gate (Wang et al., 2016)	72.58	73.94
IARNN-Occam (Wang et al., 2016)	73.41	74.18
CA-Network (Wang and Jiang, 2016)	74.33	75.45
IWAN-Skip (Shen et al., 2017b)	73.30	75.00
Dynamic-Clip (Bian et al., 2017)	75.40	76.40
WEHM (Proposed)	<b>77.02</b>	<b>78.82</b>

Table 3: Experimental results on the WikiQA dataset

Model	MAP	MRR
CNN-Cnt (Jurczyk et al., 2016)	84.00	84.94
CNN-Tree (Jurczyk et al., 2016)	84.66	85.68
RNN: one-way (Jurczyk et al., 2016)	82.06	83.18
RNN: attn-pool (Jurczyk et al., 2016)	86.43	87.59
CNN-DAN (dos Santos et al., 2017)	86.55	87.30
CNN-hinge (dos Santos et al., 2017)	87.58	88.12
AdaQA (Shen et al., 2017a)	89.14	89.83
WEHM (Proposed)	<b>91.71</b>	<b>92.22</b>

Table 4: Experimental results on the SelQA dataset

a learning rate of 0.001. Our models are trained in mini-batches (with a batch size of 10). We fix the length of the question and each sentence in the document according to their sentence’s max length in each mini-batch, and any sentences not enough to this range are padded. The hidden vector size is set to 150 for a single RNN. We conduct word sense disambiguation for ambiguous words via the nltk tool.

### 3.3 Results and Analysis

#### 3.3.1 Performance

We compare our model with state-of-the-art methods on the WikiQA and SelQA dataset in Table 3 and Table 4, respectively. Our proposed model not only obtains state-of-the-art performance on two datasets but also makes a significant improvement. Compared with the existing best method Dynamic-Clip, our model yields nearly 1.6% improvement in MAP and 2.4% in MRR on the WikiQA dataset. On the SelQA dataset, our model improves 2.6% in MAP and 2.4% in MRR, compared with the previous best method AdaQA. It is a challenging task for answer selection, especially for the WikiQA dataset. As is shown in Table 3, the notable CA-network outperforms the IARNN-Occam approach only by 0.92 MAP points, and our best result (77.02) achieves a performance gain of 1.6 MAP points over the Dynamic-Clip. In this sense, the improvement of our model is valuable.

Model	MAP	$\Delta/\%$
<b>without WordNet knowledge</b>	<b>84.87</b>	-
(1) only hypernym token	85.35	0.48
(2) only synset token	85.17	0.30
(3) only hypernym&synset token	86.32	1.45
(4) only hypernym attention	90.21	5.34
(5) only synset attention	89.99	5.12
(6) only hypernym&synset attention	90.49	5.62
<b>WEHM</b>	<b>91.71</b>	<b>6.84</b>

Table 5: Ablation study on the SelQA dataset

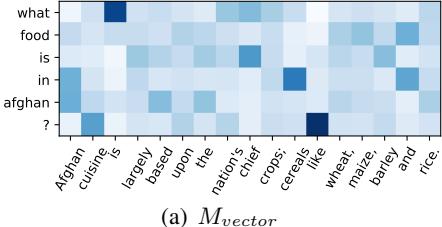
#### 3.3.2 Ablation Study

We further conduct an ablation study to explore different WordNet-enhanced components in our model, including WordNet-enhanced word embedding and WordNet-enhanced Attention Mechanism. Table 5 reports the experimental results.

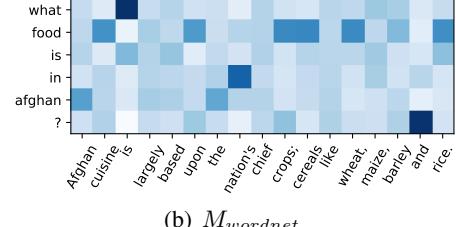
We first remove all knowledge components from our model, denoted as *without WordNet knowledge*, which can be regarded as the baseline model. In the baseline model, we only use the original word embeddings and the conventional Luong attention mechanism. Then we evaluate the WordNet-enhanced word embedding by adding the hypernym, synset, and the combination of both to the word embeddings, shown in (1)-(3) of Table 5. To evaluate the WordNet-enhanced attention mechanism, we also add the synset relation score, the hypernym score or its combination to the original hidden vectors’ score based on the baseline model, shown in (4)-(6) of Table 5.

Compared with the baseline model, the WordNet knowledge brings consistent performance gain both for the WorNet-enhanced word embedding and WordNet-enhanced attention mechanism. As for the Knowledge-enhanced word embedding, the hypernym and synset improve 0.48% and 0.30% in MAP, respectively, and the combination of them improves 1.45% in MAP. As for the Knowledge-enhanced attention mechanism, the hypernym and synset improve 5.34% and 5.12% in MAP respectively, and the combination of them improves 5.62% in MAP. At the result, our full proposed model WEHM yields a significant performance gain of 6.84 MAP points.

We could find that the knowledge-enhanced attention mechanism is more effective than the simple knowledge-enriched word embedding, perhaps because computing the similarity scores of two concepts takes into account much information, like the shortest path between them and the depth of the concept in the taxonomy. Moreover, the combina-



(a)  $M_{vector}$



(b)  $M_{wordnet}$

Figure 3: Attention score matrixes  $M_{vector}$  and  $M_{wordnet}$  of a real case on the WikiQA dataset. The matrix  $M_{wordnet}$  not only captures the paraphrase information like "food" and 'cuisine', but also enhances relations between the question's word "food" and some of the sentence's words, like "crops", "cereals", "wheat" and "rice".

tion of hypernym and synset is better than the single hypernym or synset information in both knowledge components because it captures more diverse information. Interestingly, the hypernym information is more effective than the synset information in the question-answering task.

### 3.3.3 Case Study

To make a detailed analysis of the effectiveness of our proposed model, we give a case study to visualize the different attention score matrix  $M_{vector}$  and  $M_{wordnet}$ , by a heatmap in Figure 3.  $M_{vector}$  is only computed by hidden vectors, and  $M_{wordnet}$  is calculated by our proposed model. When answering the question, our proposed model not only captures the information of "food" and "afghan", but also pays more attention to the related meaning of "wheat - food", "rice - food" and so on , which brings vital information to the prediction, while the baseline method performs weakly on capturing this information.

### 3.3.4 Error Analysis

We further make an error analysis of our model for further improvements. Table 6 is a wrong prediction produced by our proposed model (WEHM). "Cardiovascular disease" is another name for "heart disease". However, "Cardiovascular disease" isn't mentioned in the given question. Although we have enriched the model with WordNet knowledge, it is still hard for the model to capture the lexical gap between these two words, for that their concepts are not the same in WordNet. From this analysis, we'd like to employ more fine-grained knowledge, like the clarification for proper nouns.

### 3.3.5 Comparison with other knowledge-enhanced models

To the best of our knowledge, we are the first to explore the WordNet knowledge to enhance the

**Question:** what causes heart disease?

**Document:**

[1] Cardiovascular disease (also called heart disease) is a class of diseases that involve the heart or blood vessels (arteries, capillaries, and veins).

[2] .....

[3] The causes of cardiovascular disease are diverse but atherosclerosis and hypertension are the most common.

[4] .....

**Reference Answer:**

The causes of cardiovascular disease are diverse but atherosclerosis and hypertension are the most common.

Table 6: The error prediction of our proposed model. The text is shown in its original form, which may contain errors in typing. Our proposed model predict the first sentence is the right answer, however it is wrong.

neural network model for the DQA problem. There are also some other knowledge-enhanced models designed for specific tasks, in which the natural language inference (NLI) task is somewhat similar to the QA task. In order to compare with our proposed WEHM model, we re-run the KEM model on the WikiQA dataset by using its public codes, which is designed for NLI task by Chen et al. (2018). ESIM (Chen et al., 2017) is the basic model of KEM without knowledge. KEM uses feature vectors of specific dimensions in WordNet, while our WEHM model directly employs synset and hypernym relation scores to enrich the attention score and also use their concepts to enrich the word representation. Table 7 shows the results of the WikiQA dataset. We could see that our proposed model outperforms the KEM model by a large margin. Besides, when comparing the improvements produced by the enriched knowledge, our proposed model is still better than KEM, with nearly 4% gain versus about 3% gain in MAP.

Model	MAP	MRR
ESIM ( <a href="#">Lan and Xu, 2018</a> )	65.20	66.40
KEM ( <a href="#">Chen et al., 2018</a> )	68.03	69.58
WEHM (without knowledge)	73.17	74.63
WEHM (Proposed)	<b>77.02</b>	<b>78.82</b>

Table 7: Experimental results on the WikiQA dataset. We list the reported results of ESIM in the paper ([Lan and Xu, 2018](#)), and re-run the public code of KEM proposed in the paper ([Chen et al., 2018](#)) to produce its results.

## 4 Related Work

In the NLP field, many problems involve matching two or more sequences to make a decision. For the DQA task, most of the studies also consider this problem as text matching, and they compute the semantic similarity between the question and candidate answers to decide whether a sentence in the document could answer the question.

There have been various deep neural network models proposed to tackle sentence pairs matching. Two kinds of matching strategies have been considered: the first is to convert the whole source and target sentences into embedding vectors in the latent spaces respectively, and then calculate the similarity score between them; the second is to calculate the similarities among all possible local positions of the source and target sentences and then summarize the local scores into the final similarity value. As for works using the first strategy, [Qiu and Huang \(2015\)](#) apply a tensor transformation layer on CNN-based embeddings to capture the interactions between the question and answer. [Tan et al. \(2015\)](#) employ the long short-term memory (LSTM) network to address this problem. In the second strategy, [Pang et al. \(2016\)](#) build hierarchical convolution layers on the word similarity matrix between sentences, and [Yin and Schütze \(2015\)](#) propose MultiGranCNN to integrate multiple granularity levels of matching models.

For the DQA task, the notable work is the compare-aggregate structure, which is first proposed by [Wang and Jiang \(2016\)](#). Following this structure, [Bian et al. \(2017\)](#) propose the dynamic-clip way to compute the attention score. Our basic model also adopts this structure, but with a different implementation. What’s more, we employ a hierarchical module to capture inter-sentence relations.

Exploiting the background knowledge and common sense to improve NLP tasks’ performance

has long been a heated research topic. To facilitate NLP tasks, various semantic knowledge bases (KBs) have been constructed, ranging from manually annotated semantic networks like WordNet ([Fellbaum, 1998](#)) to semi-automatically or automatically constructed knowledge graphs like Freebase ([Bollacker et al., 2008](#)). Recently, several approaches have been proposed to leverage the prior knowledge in neural networks on different tasks ([Yang and Mitchell, 2017; Chen et al., 2018; Wu et al., 2018; Wang et al., 2019](#)). [Wu et al. \(2018\)](#) fuse the prior knowledge into word representations with a knowledge gate by using question categories for the QA task and topics for the conversation task. [Yang and Mitchell \(2017\)](#) propose a KBLSTM network architecture, which incorporates the background knowledge into LSTM to improve machine reading. Unlike the two approaches, our model directly employs the synset and hypernym concepts information to enrich the word representation. [Chen et al. \(2018\)](#) use WordNet to measure the semantic relatedness of word pairs for the natural language inference task, including synonym, antonym, hypernym, and same hypernym. Each of these features is denoted as a real number and is incorporated into the neural networks. Compared to the feature vectors derived from the WordNet, our model directly employ the synset and hypernym relation scores to enrich the attention mechanism. [Wang et al. \(2019\)](#) present an entailment model for solving the Natural Language Inference (NLI) problem that utilizes ConceptNet as an external knowledge source, while our method mainly focus on the WordNet.

## 5 Conclusion

In this paper, we exploit a WordNet-enhanced hierarchical model to address the answer selection problem. Based on WordNet’s prior knowledge, the proposed model applies the synset and hypernym concepts to enrich word representations and uses synset and hypernym relation scores between two concepts to enhance the traditional attention score. Extensive experiments conducted on two benchmark datasets demonstrate that our method significantly improves the baseline model and outperforms state-of-the-art results by a large margin. Our approach obtains 1.62% improvement and 2.57% improvement in MAP on the WikiQA and SelQA datasets, respectively, compared to the state-of-the-art results. In the future, we would like to

explore more knowledge in the neural networks to deal with different NLP tasks.

## Acknowledgments

This work is supported by the National Natural Science Foundation of China (61773026) and the Key Project of Natural Science Foundation of China (61936012).

## References

- Weijie Bian, Si Li, Zhao Yang, Guang Chen, and Zhiqing Lin. 2017. A compare-aggregate model with dynamic-clip attention for answer selection. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1987–1990. ACM.
- Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. ACM.
- Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Diana Inkpen, and Si Wei. 2018. Neural natural language inference models enhanced with external knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 2406–2417.
- Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced lstm for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1657–1668.
- Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.
- Christiane Fellbaum. 1998. *WordNet*. Wiley Online Library.
- Hua He and Jimmy J Lin. 2016. Pairwise word interaction modeling with deep neural networks for semantic similarity measurement. In *HLT-NAACL*, pages 937–948.
- Daniel Hewlett, Llion Jones, Alexandre Lacoste, et al. 2017. Accurate supervised and semi-supervised machine reading for long documents. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2001–2010.
- Tomasz Jurczyk, Michael Zhai, and Jinho D Choi. 2016. Selqa: A new benchmark for selection-based question answering. In *Tools with Artificial Intelligence (ICTAI), 2016 IEEE 28th International Conference on*, pages 820–827. IEEE.
- Wuwei Lan and Wei Xu. 2018. Neural network models for paraphrase identification, semantic textual similarity, natural language inference, and question answering. *arXiv preprint arXiv:1806.04330*.
- Claudia Leacock and Martin Chodorow. 1998. Combining local context and wordnet similarity for word sense identification. *WordNet: An electronic lexical database*, 49(2):265–283.
- Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.
- Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2016. Text matching as image recognition. In *AAAI*, pages 2793–2799.
- Ted Pedersen, Siddharth Patwardhan, and Jason Michelizzi. 2004. Wordnet:: Similarity: measuring the relatedness of concepts. In *Demonstration papers at HLT-NAACL 2004*, pages 38–41. Association for Computational Linguistics.
- Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.
- Xipeng Qiu and Xuanjing Huang. 2015. Convolutional neural tensor network architecture for community-based question answering. In *IJCAI*, pages 1305–1311.
- Cicero Nogueira dos Santos, Kahini Wadhawan, and Bowen Zhou. 2017. Learning loss functions for semi-supervised learning via discriminative adversarial networks. *arxiv preprint arXiv:1707.02198*.
- Dinghan Shen, Martin Renqiang Min, Yitong Li, and Lawrence Carin. 2017a. Adaptive convolutional filter generation for natural language understanding. *arXiv preprint arXiv:1709.08294*.
- Gehui Shen, Yunlun Yang, and Zhi-Hong Deng. 2017b. Inter-weighted alignment network for sentence pair modeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1179–1189.
- Ming Tan, Cicero dos Santos, Bing Xiang, and Bowen Zhou. 2015. Lstm-based deep learning models for non-factoid answer selection. *arXiv preprint arXiv:1511.04108*.
- Bingning Wang, Kang Liu, and Jun Zhao. 2016. Inner attention based recurrent neural networks for answer selection. In *ACL (1)*.

- Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. 2017a. Irgan: A minimax game for unifying generative and discriminative information retrieval models. *arXiv preprint arXiv:1705.10513*.
- Shuohang Wang and Jing Jiang. 2016. A compare-aggregate model for matching text sequences. *arXiv preprint arXiv:1611.01747*.
- Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017b. Gated self-matching networks for reading comprehension and question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 189–198.
- Xiaoyan Wang, Pavan Kapanipathi, Ryan Musa, Mo Yu, Kartik Talamadupula, Ibrahim Abdelaziz, Maria Chang, Achille Fokoue, Bassem Makni, Nicholas Mattei, et al. 2019. Improving natural language inference using external knowledge in the science questions domain. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7208–7215.
- Dirk Weissenborn, Georg Wiese, and Laura Seiffe. 2017. Making neural qa as simple as possible but not simpler. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 271–280.
- Yu Wu, Wei Wu, Can Xu, and Zhoujun Li. 2018. Knowledge enhanced hybrid neural network for text matching. In *The Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*, pages 5586–5593.
- Zhibiao Wu and Martha Palmer. 1994. Verbs semantics and lexical selection. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, pages 133–138. Association for Computational Linguistics.
- Bishan Yang and Tom Mitchell. 2017. Leveraging knowledge bases in lstms for improving machine reading. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1436–1446.
- Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. Wikiqa: A challenge dataset for open-domain question answering. In *EMNLP*, pages 2013–2018.
- Wen-tau Yih, Ming-Wei Chang, Christopher Meek, and Andrzej Pastusiak. 2013. Question answering using enhanced lexical semantic models. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1744–1753.
- Wenpeng Yin and Hinrich Schütze. 2015. Multi-grancnn: An architecture for general matching of text chunks on multiple levels of granularity. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 63–73.
- Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. 2015. Abcnn: Attention-based convolutional neural network for modeling sentence pairs. *arXiv preprint arXiv:1512.05193*.

# A Simple Text-based Relevant Location Prediction Method using Knowledge Base

Mei Sasaki

{mesasaki, sokura, shiono}@yahoo-corp.jp

Yahoo Japan Corporation

Tokyo, Japan

Shumpei Okura

Shingo Ono

## Abstract

In this paper, we propose a simple method to predict salient locations from news article text using a knowledge base (KB). The proposed method uses a dictionary of locations created from the KB to identify occurrences of locations in the text and uses the hierarchical information between entities in the KB for assigning appropriate saliency scores to regions. It allows prediction at arbitrary region units and has only a few hyperparameters that need to be tuned. We show using manually annotated news articles that the proposed method improves the f-measure by  $> 0.12$  compared to multiple baselines.

## 1 Introduction

Predicting relevant locations from news articles can result in numerous useful applications. For example, it enables the delivery of news related to a specific city that is of user interest, or facilitates the prediction of a disease outbreak in a specific region when used with event detection techniques.

In this paper, we focus on predicting relevant locations from news articles. The goal of this task is to identify locations that are salient to the article, not those that simply appeared in the article. For example, consider the following excerpt: “The Aoi Festival is one of the three major festivals in Kyoto. It originated as a series of rites to calm down angry gods. A visitor from Australia said...” In this example, the phrase “Kyoto” is highly relevant to the article, but “Australia” is not.

Traditional methods to predict locations require specific data, such as a training dataset or phrase distribution, that match the application domain and the granularity of the prediction. However, it is costly to prepare such data for individual applications.

We propose a knowledge base (KB)-based method that only requires a general-purpose KB

instead of a labeled dataset for training. It propagates phrase-level importance to region entities following their relationship in the KB. It can theoretically be applied to predictions at an arbitrary level of granularity (e.g. countries, prefectures, cities) without dedicated training data.

In this study, we focus on Japanese news articles and report the performance of predictions at the Japanese prefecture-level. We provide practical tips to tackle un-tokenized language like Japanese.

## 2 Related Works

Depending on the objective, geolocation prediction tasks from texts are roughly divided into two types. One type is for detecting and identifying mentions of points-of-interest (POIs) in the text, well known by entity linking (Nadeau and Sekine, 2007; Shen et al., 2015). This task focuses on extracting all mentions regardless of their saliency. The other type is for estimating the author’s current location or home town from his or her posts (Huang and Carley, 2019). It is mainly performed to complement user profiles in services that deal with user-generated content (e.g. SNS). In this case, in addition to text, various user metadata such as the relationship between authors is available (Backstrom et al., 2010). Our work is similar to the former type in terms of purpose, but we focus on identifying salient regions rather than extracting all of the individual POIs.

There are two main approaches to predicting location. One is the dictionary-based approach (Berggren et al., 2015; Han et al., 2012; Li et al., 2014), where dictionaries of location indicative words are created in advance and used for the prediction. In addition to explicit region names, the choice of which words to add to the dictionary is a hot topic of discussion (Han et al., 2014). The other is the machine learning(ML)-based ap-

proach (Zhou and Luo, 2012; Miyazaki et al., 2018). Methods based on this approach usually perform well if sufficient training data is available. However, in practice, it is difficult to prepare data whose granularity matches the requirements of the application. In particular, estimating regions that are rarely found in the training data is one of the weaknesses of machine learning.

### 3 Task Setting and Baseline

#### 3.1 Task

Let  $A$  be the set of articles and  $R$  be the set of candidate regions, e.g., Japanese prefectures. Our goal is to construct a function  $\mathcal{F} : A \rightarrow \mathfrak{P}(R)$  such that  $r \in \mathcal{F}(a)$  if and only if there are mentions of region  $r$  in the article  $a$  and region  $r$  is salient to the text, where  $\mathfrak{P}(R)$  denotes the power set of  $R$ . Note that even when there are mentions of  $r$  in  $a$ , if  $r$  is not a main topic in  $a$ ,  $\mathcal{F}(a)$  does not contain  $r$ . Similarly, when  $a$  is not a location-aware article, then  $\mathcal{F}(a)$  is  $\emptyset$ . In this paper, we assume  $A$  to be a set of Japanese news articles and  $R$  to be a set of Japanese prefectures.

#### 3.2 Gazetteer baseline

The baseline method we adopted is similar to the baseline used in (Berggren et al., 2015) and consists of the following three steps:

- 1) Create a gazetteer from an external data source.
- 2) Identify the strings contained in the gazetteer from the given text.
- 3) Aggregate the results and return the relevant locations.

In practice, there are several choices regarding step 3). For example, we could return all the regions mentioned in the text, return the region mentioned most frequently in the text, or return only the region that appears earliest in the text. We decided to return locations that appear in the first 20% of the text.

## 4 KB-based Methods

#### 4.1 Knowledge base

A KB consists of information about entities expressed in a structured, machine-readable graph format. YAGO and DBpedia are examples of KBs (Hoffart et al., 2011; Lehmann et al., 2015). Entities are assigned class(es) that represent what kind of entity they are. Examples of possible entity classes include person, place, and company. Mount Fuji is an example of a place entity.

A KB can be regarded as an edge-labeled directed graph. Entities correspond to nodes in the graph, and the relationships between entities correspond to edges in the graph. These relationships are given relation-type labels called *predicates*.

We focus on the subgraph of KB that is useful in terms of location prediction. Let us reduce the graph by keeping only *Place* class entities, and vertexes connected to such entities with inclusive relations such as *containedBy* predicates and notation relations such as *name*, *alsoKnownAs* predicates, and name the resulting graph  $G = (V, E)$ . The notation relations in the graph will be used in §4.3 to create a gazetteer, while the inclusive relations will be used in §4.5 and §4.6 to determine the set of corresponding entities for each mention and calculate the corresponding score for different candidate regions, respectively.

#### 4.2 Overview of the proposed method

The overview of our proposed method is shown in Fig. 1. It consists of four steps, three of which correspond to those in §3.2 and the rest is the entity linking step performed between 2) and 3). As shown in the following sections, we add the efficient use of KB information at each step.

#### 4.3 Create gazetteer

Create a dictionary  $\mathcal{D}$  with location names as keys and corresponding entities as values using the notation relations in the KB. Note that multiple entities may have the same name, so  $\mathcal{D}(m)$  is a set of entities that belong to  $V$  for each key  $m$ .

However, in practice, if we use all of the notation relations for  $\mathcal{D}$ , it may adversely affect the prediction. For example, “the park” can be an alias for all parks in the world, but due to some inconsistencies in KB entries, some parks have such an alias in the KB and others do not. Therefore, by using inclusive relations between entities in the KB, we systematically extract phrases that appear as the prefix/postfix of entity names in wildly distant multiple regions and create a blacklist of phrases by manually reviewing them. In addition, we manually added the names of central ministries to the blacklist, since occurrences of such names rarely indicate the locality of the news. The blacklist currently consists of 151 words.

#### 4.4 Phrase identification

When given an article  $a$ , identify phrases that serve as clues by the following three steps:

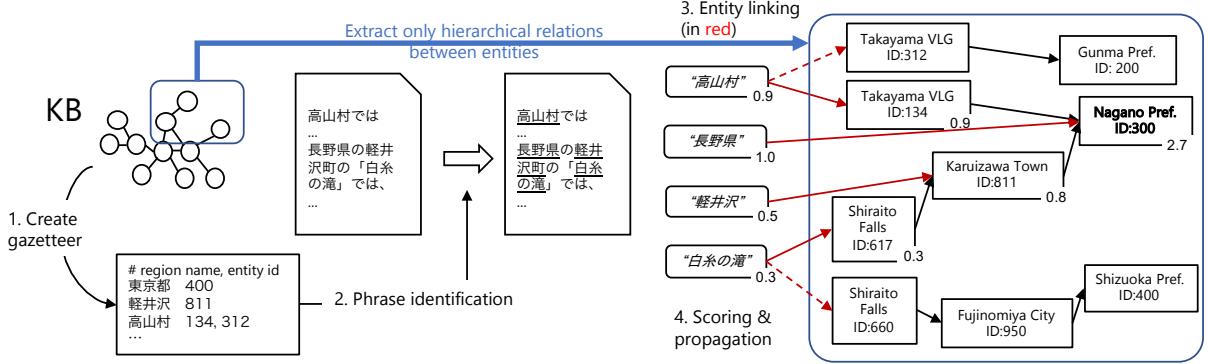


Figure 1: Overview of the proposed method.

1. Tokenize text of the article into morphemes.
2. Chunk the list of morphemes so that it results in as long and as many matches for keys in  $\mathcal{D}$ .
3. Perform named entity recognition (NER) and only keep phrases that at least partially overlap with named entities whose IREX<sup>1</sup> category is *LOCATION*, *ARTIFACT*, or *ORGANIZATION*.

The additional NER step is necessary to avoid confusion between the names of persons and places. There are many family names in Japanese that have similar characters to region names.

The reason we do not limit the phrases to those categorized as *LOCATION* is that some *ORGANIZATION* or *ARTIFACT* entities may contain region names in their names, e.g., small local businesses. The sets of phrases identified from article  $a_i$  in this step are represented by  $\{m_i\}$  hereafter.

#### 4.5 Entity linking

Entity linking is the task of mapping entity mentions to the corresponding entities in a KB. The purpose of this step is to reduce the candidate entities for  $m_i$  from  $\mathcal{D}(m_i)$  using the contexts of article  $a$ .  $\mathcal{D}_a(m_i)$  denotes the candidates remaining after the following steps.

1. If  $|\mathcal{D}(m_i)| = 1$ , we have nothing to do.
2. If  $\mathcal{D}(m_i)$  contains  $e$  s.t.  $\exists m_j \text{ in } a, \mathcal{D}(m_j) = \{e\}$ , we remove other candidates for  $m_i$ .
3. If  $\mathcal{D}(m_i)$  doesn't satisfy the above but contains  $e$  s.t.  $\exists m_j \text{ in } a, \mathcal{D}(m_j) = \{e'\}$ ,  $e$  and  $e'$

<sup>1</sup><https://nlp.cs.nyu.edu/irex/NE/df990214.txt>

are both contained by the same region  $r \in R$ , we remove other candidates for  $m_i$ .

In short, we give preference to entities when there is relevant evidence elsewhere in the article.

Unlike in traditional entity linking tasks, for our purposes, if the procedure fails to resolve the phrase to a single entity, but finds a list of candidates, it is still quite useful in terms of location prediction. As discussed later in the paper, such phrases and corresponding candidate entities will be taken properly into account in the later steps.

#### 4.6 Scoring and propagation

In this step, we score each phrase occurrence  $m_i$  and propagate the score to corresponding entities.

First, we define phrase scores  $\phi_a$  as:

$$\phi_a(m_i) = \frac{\text{length}(m_i)}{\log(\text{pos}(m_i) + C)},$$

where  $\text{pos}(m_i)$  means the number of words that precede  $m_i$  in the article and  $C$  is a positive constant.

Next, we calculate entity scores  $\psi_a$  as:

$$\psi_a(e_i) = \sum_{(e_j, e_i) \in E} \frac{\phi_a(e_j)}{|\{(e_j, \cdot) \in E\}|} + \sum_{e_i \in \mathcal{D}_a(m_j)} \frac{\phi_a(m_j)}{|\mathcal{D}_a(m_j)|},$$

where  $E$  has a DAG structure because it is composed of inclusive relations and the calculation order is naturally determined.

Finally, return regions that satisfy certain criteria as  $\mathcal{F}(a)$ . There are several possibilities for the actual criteria to determine which regions to return, such as

$$\begin{aligned} \mathcal{F}(a) &= \{r \in R | \psi_a(r) > T\} \text{ (absolute),} \\ \mathcal{F}(a) &= \{r \in R | \frac{\psi_a(r)}{\max_{r'}(\psi_a(r'))} > \alpha\} \text{ (relative),} \\ \mathcal{F}(a) &= \{r \in R | \text{rank}(\psi_a(r)) \leq N\} \text{ (rank).} \end{aligned}$$

# of articles	1,711
# of candidate prefs	47
# of salient prefs / article	1.29
articles with no salient prefs	28.4%

Table 1: Statistics on dataset.

After experiments, we decided to adopt the intersection of the above three criteria with parameters  $T = 0.5$ ,  $\alpha = 0.7$ , and  $N = 2$ .

## 5 Experiments

### 5.1 Knowledge base resource

We implemented the method proposed in §4 using an in-house KB of Yahoo Japan Corporation ([Yamazaki et al., 2019](#)) and in-house morphological analysis/NER tools for the following experiments. In short, the KB consists of data integrated from various open data, data purchased from our suppliers, and information extracted from web crawling. When open data is available in multiple languages (e.g., Wikipedia), a Japanese data dump is used to construct the KB. For historical reasons, the entities that correspond to regions in the KB were little used, and there were problems regarding the quality of data in this domain. Therefore, we incorporated various official data sources containing lists of regions, regional codes, and zip codes into the KB, as the accuracy/completeness of regions and the inclusive relations between them play a crucial role in location prediction.

### 5.2 Dataset

Since there is no publicly available Japanese corpus of salient locations, we asked a team of professional annotators to label a total of 1,711 news articles with relevant prefectures. There are 47 prefectures in Japan. The details of this dataset are shown in Table 1. The team consists of five annotators independent of us. Although each article is labeled by one annotator, the annotation team created an annotation guideline in an iterative way as follows to ensure consistency of the annotation:

First, create a temporary annotation guideline and annotate a relatively small number of articles. Then, share the annotated results and discuss whether an annotation guideline needs to be updated. This iteration was repeated until a reasonable annotation guideline is fixed. Note that the annotation guideline was finalized before the development of the proposed method started.

Although our method enables prediction with finer granularity, we evaluate only at the prefecture-level in this first research due to the cost of annotation.

### 5.3 Metrics

We evaluate the prediction performance by micro-averaged precision ( $p_m$ ), micro-averaged recall ( $r_m$ ), and article-averaged f-measure ( $f_A$ ) calculated as:

$$p_m = \frac{\sum_{a \in A} |R_a \cap \mathcal{F}(a)|}{\sum_{a \in A} |\mathcal{F}(a)|}, f_A = \frac{1}{|A|} \sum_{a \in A} \frac{2p_a r_a}{p_a + r_a},$$

$$r_m = \frac{\sum_{a \in A} |R_a \cap \mathcal{F}(a)|}{\sum_{a \in A} |R_a|},$$

where  $R_a$  is the set of salient prefectures for article  $a$  in the ground truth and  $p_a = |R_a \cap \mathcal{F}(a)|/|\mathcal{F}(a)|$ ,  $r_a = |R_a \cap \mathcal{F}(a)|/|R_a|$  are article-level precision and recall, respectively. We consider  $p_a = 1.0$  if  $\mathcal{F}(a) = \emptyset$ , and  $r_a = 1.0$  if  $R_a = \emptyset$ . Hence, when  $a$  is not a location-aware article, the harmonic average of the two is 1.0 if and only if the method returns an empty set, and 0.0 otherwise.

### 5.4 Baseline methods

We adopted two different baseline methods to demonstrate the validity of the proposed method, the baseline method that relies on gazetteer described in 3.2 and ML-based method.

The second ML-based baseline method treats location prediction as a multi-label classification problem (i.e., an article can have multiple subject regions). In this setting, the classifier assigns different labels that correspond to Japanese prefectures to each article. We used fastText ([Joulin et al., 2017](#)) library for this task and tokenized the text for each article using the same in-house morphological tool described in 5.1.

### 5.5 Comparison with baselines

The results for the proposed and baseline methods are listed in Table 2. As shown, the proposed method outperformed the baseline methods in all performance metrics. Note that the evaluation metrics for fastText baseline are calculated in a slightly different way from other methods and are meant as approximate reference values. It was calculated by taking an average of models obtained

methods	$p_m$	$r_m$	$f_A$
gazetteer baseline	0.501	0.476	0.708
fastText baseline	0.660*	0.420*	0.430*
proposed	0.824	<b>0.515</b>	0.830
proposed + BL	<b>0.856</b>	<b>0.515</b>	<b>0.852</b>

Table 2: Results of proposed and baseline methods.

\* The average over nested 4-fold cross-validation.

by nested 4-fold cross-validation over the evaluation dataset. We tuned the hyperparameters to optimize article-averaged f-measure ( $f_A$ ) in each inner loop of cross-validation. For the proposed method and the gazetteer baseline that require no dataset for training, the evaluation metrics are calculated using the entire dataset.

The gazetteer baseline suffers from low precision. We give the following example to demonstrate how the proposed method’s output improved over that of the gazetteer baseline. Yokohama most often represented the well-known city in Kanagawa Pref. but on rare occasions represented the small town with a similar name in Aomori Pref. The gazetteer baseline is not able to prioritize between them and returns both locations. The proposed method considered the other entity mentions to resolve Yokohama into the correct region.

The fastText baseline performs differently for different kinds of articles. While most of the articles in the evaluation dataset are labeled one or two prefectures, some articles contain phrases that collectively refer to multiple Japanese prefectures<sup>2</sup> and are labeled a large number of prefectures. Since such phrases have only a limited number of variations and appear in the dataset repeatedly, it is relatively easy for the ML-based approach to learn such expressions. Therefore it performs relatively well in terms of micro-averaged metrics heavily weighted to articles with a high number of relevant prefectures. However, the article-averaged metric  $f_A$  is incredibly low compared to other methods. This is because the knowledge of names of individual prefectures or cities is essential in order to make correct predictions for the rest of the articles. We found that fastText classifier often fails to predict locations for such articles even when names of prefectures are explicitly stated in the article. We conclude that it is practically impossible to learn all the necessary region names from a few thou-

<sup>2</sup>Examples include “Tōhoku region” that refers to 6 prefectures and “Western Japan” that refers to > 20 prefectures.

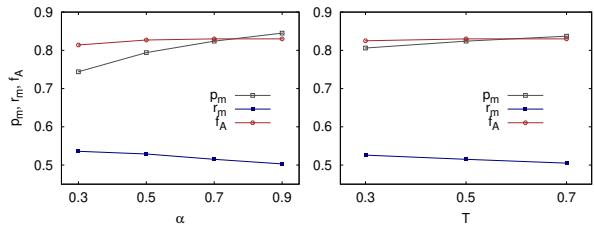


Figure 2: Effect of varying  $\alpha$  (left) and  $T$  (right).

sand articles and that utilizing external resources such as the KB is the critical element in achieving good performance.

As another example, there is “Tokyo Disney-land” that is actually located in Chiba Pref., not in Tokyo Pref. It is crucial to treat it as an entity and not be confused by their apparent region names.

When we added the blacklist created in §4.3 to the proposed method, there was a huge improvement in precision. This highlights the incompleteness of the aliases in the KB and indicates that care must be taken when applying entries in KB to a real service.

## 5.6 Impact of hyperparameters

The hyperparameters that govern the performance of our proposed method are  $T$ ,  $\alpha$  and  $N$  (introduced in §4.6). We can see the effect of varying these hyperparameters in Fig. 2. These results demonstrate that the precision/recall tradeoff can be adjusted by varying hyperparameters.

## 6 Conclusion

In this paper, we presented a simple KB-based method to predict relevant locations from articles. The proposed method requires no training data or maintenance of a dictionary thanks to a freshly generated KB, and it can be used to make predictions at an arbitrary level of granularity, as long as the corresponding data is present in the KB. We demonstrated the effectiveness of this method at predicting salient Japanese prefectures using manually annotated articles. In future work, we plan to make location predictions at the city-level and evaluate its performance.

## Acknowledgments

We thank our teammates for help in developing and deploying our location prediction system. We are grateful to the annotation team for providing us with an evaluation dataset. We wish to thank the anonymous referees for helpful comments.

## References

- Lars Backstrom, Eric Sun, and Cameron Marlow. 2010. [Find me if you can: improving geographical prediction with social and spatial proximity](#). In *Proceedings of the 19th International Conference on World Wide Web*, pages 61–70.
- Max Berggren, Jussi Karlgren, Robert Östling, and Mikael Parkvall. 2015. [Inferring the location of authors from words in their texts](#). In *Proceedings of the 20th Nordic Conference of Computational Linguistics*, pages 211–218.
- Bo Han, Paul Cook, and Timothy Baldwin. 2012. [Geolocation prediction in social media data by finding location indicative words](#). In *Proceedings of COLING 2012*, pages 1045–1062.
- Bo Han, Paul Cook, and Timothy Baldwin. 2014. [Text-based twitter user geolocation prediction](#). *Journal of Artificial Intelligence Research*, 49(1):451–500.
- Johannes Hoffart, Fabian M Suchanek, Klaus Berberich, Edwin Lewis-Kelham, Gerard De Melo, and Gerhard Weikum. 2011. [Yago2: exploring and querying world knowledge in time, space, context, and many languages](#). In *Proceedings of the 20th International Conference Companion on World Wide Web*, pages 229–232.
- Binxuan Huang and Kathleen Carley. 2019. [A hierarchical location prediction neural network for twitter user geolocation](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 4732–4742.
- Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. [Bag of tricks for efficient text classification](#). In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431.
- Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. [Dbpedia—a large-scale, multilingual knowledge base extracted from wikipedia](#). *Semantic Web*, 6(2):167–195.
- Guoliang Li, Jun Hu, Jianhua Feng, and Kian-lee Tan. 2014. [Effective location identification from microblogs](#). In *2014 IEEE 30th International Conference on Data Engineering*, pages 880–891.
- Taro Miyazaki, Afshin Rahimi, Trevor Cohn, and Timothy Baldwin. 2018. [Twitter geolocation using knowledge-based methods](#). In *Proceedings of the 2018 EMNLP Workshop W-NUT: The 4th Workshop on Noisy User-generated Text*, pages 7–16.
- David Nadeau and Satoshi Sekine. 2007. [A survey of named entity recognition and classification](#). *Lingvisticae Investigationes*, 30(1):3–26.
- Wei Shen, Jianyong Wang, and Jiawei Han. 2015. [Entity linking with a knowledge base: Issues, techniques, and solutions](#). *IEEE Transactions on Knowledge and Data Engineering*, 27(2):443–460.
- Tomoya Yamazaki, Kentaro Nishi, Takuya Makabe, Mei Sasaki, Chihiro Nishimoto, Hiroki Iwasawa, Masaki Noguchi, and Yukihiro Tagami. 2019. [A scalable and plug-in based system to construct a production-level knowledge base](#). In *D2KG@KDD*.
- Youjie Zhou and Jiebo Luo. 2012. [Geo-location inference on news articles via multimodal plsa](#). In *Proceedings of the 20th ACM International Conference on Multimedia*, pages 741–744.

# Learning Goal-oriented Dialogue Policy with Opposite Agent Awareness

Zheng Zhang<sup>†</sup>, Lizi Liao<sup>‡</sup>, Xiaoyan Zhu<sup>†</sup>, Tat-Seng Chua<sup>‡</sup>,  
Zitao Liu<sup>§</sup>, Yan Huang<sup>§</sup>, Minlie Huang<sup>†\*</sup>

<sup>†</sup> Department of Computer Science and Technology, Institute for Artificial Intelligence,  
State Key Lab of Intelligent Technology and Systems,

Beijing National Research Center for Information Science and Technology, Tsinghua University  
<sup>‡</sup> School of Computing, National University of Singapore, Singapore    <sup>§</sup> TAL Education Group

<sup>†</sup>zhangz.goal@gmail.com {zxy-dcs,aihuang}@tsinghua.edu.cn

<sup>‡</sup>liaolizi.llz@gmail.com chuats@comp.nus.edu.sg <sup>§</sup>{liuzitao,galehuang}@100tal.com

## Abstract

Most existing approaches for goal-oriented dialogue policy learning used reinforcement learning, which focuses on the target agent policy and simply treats the opposite agent policy as part of the environment. While in real-world scenarios, the behavior of an opposite agent often exhibits certain patterns or underlies hidden policies, which can be inferred and utilized by the target agent to facilitate its own decision making. This strategy is common in human mental simulation by first imaging a specific action and the probable results before really acting it. We therefore propose an opposite behavior aware framework for policy learning in goal-oriented dialogues. We estimate the opposite agent’s policy from its behavior and use this estimation to improve the target agent by regarding it as part of the target policy. We evaluate our model on both cooperative and competitive dialogue tasks, showing superior performance over state-of-the-art baselines.

## 1 Introduction

In goal-oriented dialogue systems, dialogue policy plays a crucial role by deciding the next action to take conditioned on the dialogue state. This problem is often formulated using reinforcement learning (RL) in which the user serves as the environment (Levin et al., 1997; Rieser and Lemon, 2011; Lemon and Pietquin, 2012; Young et al., 2013; Fatemi et al., 2016; Zhao and Eskenazi, 2016; Dhingra et al., 2016; Su et al., 2016; Li et al., 2017; Williams et al., 2017; Liu and Lane, 2017; Lipton et al., 2018; Liu et al., 2018; Gao et al., 2019; Takanobu et al., 2019, 2020; Jhunjhunwala et al., 2020). However, different from symbolic-based and simulation-based RL tasks, such as chess (Silver et al., 2016) and video games (Mnih et al.,

2015), which can get vast amounts of training interactions in low cost, dialogue systems require to learn directly from real users, which is too expensive.

Therefore, there are some efforts using simulation methods to provide an affordable training environment. One principle direction for mitigating this problem is to leverage human conversation data to build a user simulator, and then to learn the dialogue policy by making simulated interactions with the simulator (Schatzmann et al., 2006; Li et al., 2016; Gür et al., 2018).

However, there always exist discrepancies between simulated users and real users due to the inductive biases of the simulation model, which can lead to a sub-optimal dialogue policy (Dhingra et al., 2016).

Another direction is to learn the dynamics of dialogue environment during interacting with real user, and concurrently use the learned dynamics for RL planning (Peng et al., 2018; Su et al., 2018; Wu et al., 2018; Zhang et al., 2019b). Most of these works are based on Deep Dyna-Q (DDQ) framework (Sutton, 1990), where a world model is introduced to learn the dynamics (which is much like a simulated user) from real experiences. The target agent’s policy is trained using both real experiences via direct RL and simulated experiences via a world-model.

In the above methods, both the simulated user and world model facilitate target policy learning by providing more simulated experiences and remain a black box for the target agent. That is, the target agent’s knowledge about the simulated agents is still passively obtained through interaction and implicitly learned by the policy model updating as indirect try-and-error with real user. However, we argue that from the angle of a target agent, actively exploring the world with proper estimation would not only make user simulation

\*Corresponding author.

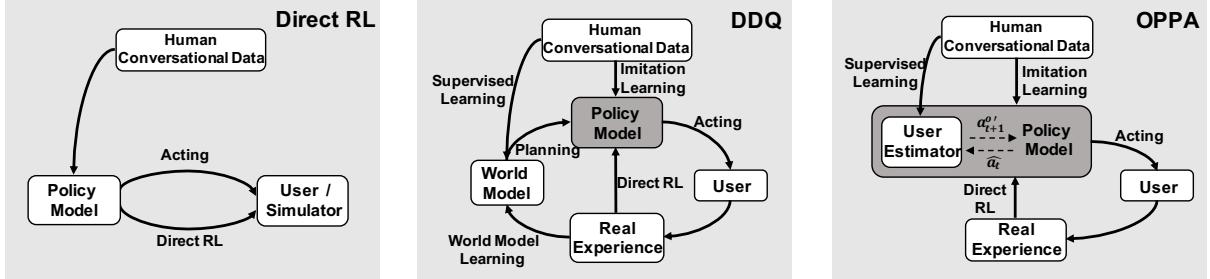


Figure 1: A comparison of dialogue policy learning a) with real/simulated user, b) with real user via DDQ and c) with real user guided by active user estimation.

more efficient but also improve the target agent’s performance. In agreement with the findings from cognitive science, humans often maintain models of other people they interact with to capture their goals (Harper, 2014; Premack and Woodruff, 1978). And humans manage to use their mental process to simulate others’ behavior (Gordon, 1986; Gallese and Goldman, 1998). Therefore, to carefully treat and model the behaviors of other agents would be full of potential. For example, in competitive tasks such as chess, the player often sees a number of moves ahead by considering the possible reaction of the other player. In goal-oriented dialogues for a hotel booking task, the agent can reduce interaction numbers and improve user experience by modeling users as business travellers with strict time limit or backpackers seeking adventure.

In this paper, we propose a new dialogue policy learning method with OPPosite agent Awareness (OPPA), where the agent maintains explicit modeling of the opposite agent or user for facilitating its own policy learning. Different from DDQ, the estimated user model is not utilized as a simulator to produce simulated experiences, but as an auxiliary component of the target agent’s policy to guide the next action. Figure 1(c) shows the framework of our model. Specifically, whenever the system needs to output an action, it foresees a candidate action  $\hat{a}_t$  and consequently estimates the user’s response behavior  $a_{t+1}^o$ . On top of this estimation, as well as the dialogue context, it makes better decisions with a dynamic estimation of the user’s strategy. To further regulate the behavior of the system agent, we mitigate the difference between the real system action  $a_t$  and the sampled action  $\hat{a}_t$  with decay for better robustness and consistency. Without any constraint on the type of agents (either competitive or cooperative), the proposed OPPA method can be applied to both cooperative and non-cooperative goal-oriented dialogues.

To summarize, our contributions are three-fold:

- We propose a new dialogue policy learning setting where the agent shifts from passively learning to actively estimating the opposite agent or user for more efficient simulations, thereby obtaining better performance.
- We mitigate the difference between real system agent action and the sampled action with decay to further enhance estimated system agent behaviors.
- Extensive experiments on both cooperative and competitive goal-oriented dialogues indicate that the proposed model can achieve better dialogue policies than baselines.

## 2 Related Work

### 2.1 RL-based Dialogue Policy Learning

Policy learning plays a central role in building goal-oriented dialogue systems by deciding the next action, which is often formulated using the RL framework. Early methods used probabilistic graph model, such as partially observable Markov decision process (POMDP), to learn dialogue policy by modeling the conditional dependences between observation, belief states and actions (Williams and Young, 2007). However, these methods require manual work to define features and state representation, which leads to poor domain adaptation. More recently, deep learning methods are applied in dialogue policy learning, including DQN (Mnih et al., 2015) and Policy Gradient (Sutton et al., 2000) methods, which mitigate the problem of domain adaptation through function approximation and representation learning (Zhao and Eskenazi, 2016).

Recently, there are some efforts focused on multi-domain dialogue policy. An intuitive way is to learn independent policies for each specific domain and aggregate them (Wang et al., 2014; Gašić

et al., 2015; Cuayahuitl et al., 2016). There are also some works using hierarchical RL, which decomposes the complex task into several sub-tasks (Peng et al., 2017; Casanueva et al., 2018) according to pre-defined domain structure and cross-domain constraints. Nevertheless, most of the above works regard the opposite agent as part of the environment without explicitly modeling its behavior.

Planning based RL methods are also introduced to make a trade-off between reducing human interaction cost and learning a more realistic simulator. Peng et al. (2018) proposed to use Deep Dynamic Q-network, in which a world model is co-trained with the target policy model. By training the world model with the real system-human interaction data, it consistently approaches the performance of real users, which provides better simulated experience for planning. Adversarial methods are applied to dynamically control the proportion of simulated and real experience during different stages of training (Su et al., 2018; Wu et al., 2018). Still, these methods work from the opposite agents' angle.

## 2.2 Dialogue User Simulation

In RL-based dialogue policy learning methods, a user simulator is often required to provide affordable training environments due to the high cost of collecting real human corpus. Agenda-based simulation (Schatzmann et al., 2007; Li et al., 2016) is a widely applied rule-based method, which starts with a randomly generated user goal that is unknown to the system. During a dialogue session, it remains a stack data structure known as *user agenda*, which holds some pending user intentions to achieve. In the stack update process, machine learning or expert-defined methods can be applied. There are also some model-based methods that learn a simulator from real conversation data. The seq2seq framework has recently been introduced by encoding dialogue history and generates the next response or dialogue action (Asri et al., 2016; Kreyssig et al., 2018). By incorporating a variational step to the seq2seq network, it can introduce meaningful diversity into the simulator (Gür et al., 2018). Our work tackles the problem from a different point of view. We let the target agent approximate an opposite agent model to save user simulation efforts.

## 3 Model

In this section, we introduce our proposed OPPA model. There are two agents in our framework, one

is the system agent we want to optimize, and the other is the user agent. We refer to these two agents as *target* and *opposite* agents in the following sections. Note that the proposed model works at dialog act level, and it can also work at natural language level when equipped with natural language understanding (NLU) and natural language generation (NLG) modules.

### 3.1 Overview

As shown in Figure 2, the proposed model consists of two key components: a target agent Q-function  $Q(s, a)$  and an opposite agent policy estimator  $\pi^o(s, a)$ . Specifically, each time before the target agent needs to take an action, the model samples a candidate action  $\hat{a}_t$ . Then the opposite estimator  $\pi_o$  estimates the opposite agent's response behavior  $a_{t+1}^o$ ', which is then aggregated with the original dialog state  $s_t$  to generate a new state  $\hat{s}_t$ . On top of this new state, the target policy  $Q(s, a)$  gets the next target action. In more detail, a brief script of our proposed OPPA model is shown in Algorithm 1.

### 3.2 Opposite Action Estimation

One essential target of the opposite estimator is to measure how the opposite agent reacts given its preceding target agent action and state. In OPPA, we implement the opposite estimation model using a two-layer feed-forward neural network followed by a softmax layer. It takes as input the current state  $s_t$ , a sampled target action  $\hat{a}_t$ , and predicts an opposite action  $a_{t+1}^o$ ' as below:

$$a_{t+1}^o' = \pi^o(s_t, \hat{a}_t). \quad (1)$$

Note that we regard the opposite action estimation task as a classification problem, and  $a_{t+1}^o$ ' is an action label. It has been shown effective in other studies like Su et al. (2018). We also carried out preliminary experiments on other more complicated designs such as Weber et al. (2017). However, results have shown MLP's superior performance in our dialogue policy learning task.

### 3.3 Opposite Aware Q-Learning

After obtaining the estimated opposite reaction  $a_{t+1}^o$ ', it serves as an extra input to the DQN-based policy component besides the original dialogue state representation  $s_t$ . Therefore, we form a new state representation  $\hat{s}_t$  as below:

$$\hat{s}_t = [s_t, E^o a_{t+1}^o'], \quad (2)$$

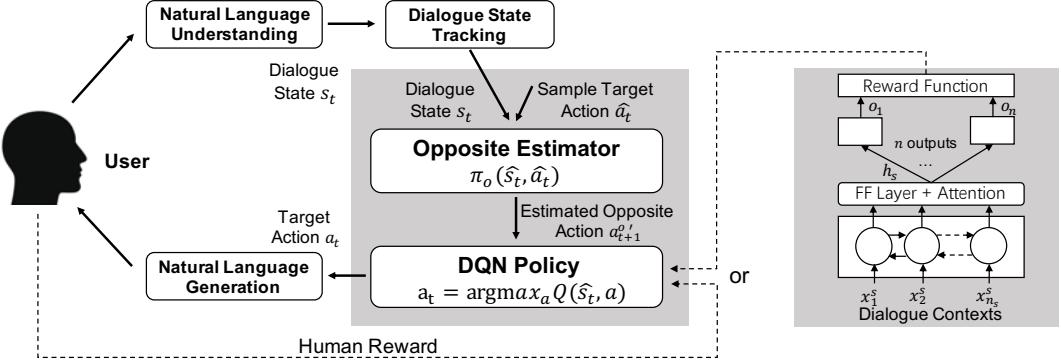


Figure 2: The proposed OPPA model and the reward function. Note that the reward for policy model can be either from real user or the reward function depending on whether real reward is available.

in which  $E^o a_{t+1}^o$ ' introduces the knowledge of opposite agent into our policy learning.  $E^o$  is the opposite action embedding matrix which maps the action into specific vector representation. Given the output  $a_t$  of  $\text{argmax}_{a'} Q(\hat{s}_t, a')$ , the agent chooses an action to execute using an  $\epsilon$ -greedy policy that selects a random action with probability  $\epsilon$  or otherwise follows the output  $a_t$ . We update the Q-function by minimizing the mean-squared loss function, which is defined as

$$\mathcal{L}_1(\theta) = \mathbb{E}_{(s,a,r,s') \sim \mathcal{D}^L} [(y_i - Q(s, a))^2], \quad (3)$$

$$y_i = r + \gamma \max_{a'} Q'(s', a'), \quad (4)$$

where  $\gamma \in [0, 1]$  is a discount factor,  $\mathcal{D}^L$  is the replay buffer and  $y_i$  represents the expected reward computed based on the transition.

### 3.4 Target Action Sampling

In this subsection, we explain how the action  $\hat{a}_t$  is sampled utilizing the above modules. For generating the true target action  $a_t$ , we predict it using a deep Q-network which takes as input an estimated opposite action  $a_{t+1}^o$ ' and the dialogue state  $h_t^e$ . However, we cannot get  $a_{t+1}^o$ ' without  $\hat{a}_t$ . Therefore, we further leverage this Q-network at hand. Specifically, we feed a constant opposite action placeholder  $a^o$  to the Q-function:

$$\hat{a}_t = \text{argmax}_{a'} Q([h_t^e, E^o a^o], a') \quad (5)$$

where  $a^o$  serves as a constant opposite action. In our experiment,  $a^o$  corresponds to the general actions which do not influence business logic, such as *Hello* and *Thanks*.

### 3.5 Action Regularization with Decay

In our method,  $\hat{a}_t$  is sampled from a distribution. At the very beginning of training, since the model is

not well trained, the sampled  $\hat{a}_t$  may perform badly, which would lead to slow convergence. Therefore, we apply action regularization to mitigate the difference between  $\hat{a}_t$  and real  $a_t$ . As the training progress goes on, such guidance becomes less effective, and we hope to encourage the model to explore more in the action space. Therefore, we adopt a decay mechanism inspired by (Zhang et al., 2019a). The regularization term is defined as the cross entropy of  $\hat{a}_t$  and real action:

$$\mathcal{L}_2(\theta) = -\beta \sum_t a_t \log(\hat{a}_t), \quad (6)$$

where  $\beta$  is the decay coefficient. The value of  $\beta$  decreases along with time by applying a discount factor  $\gamma$  in each epoch. As a consequence, a strict constraint on the sampled action is applied to avoid large action sampling performance drop at the beginning stage. After that, the constraint is continuously relaxed so that the model can explore more actions for better strategy.

To sum up, the final loss function for training our OPPA model is the weighted sum of the DQN loss and action regularization loss:

$$\mathcal{L}(\theta) = \mathcal{L}_1(\theta) + \lambda \mathcal{L}_2(\theta). \quad (7)$$

where  $\lambda$  is an adjustable hyperparameter.

### 3.6 Reward Function

When a dialogue session is completed, what we get are several dialogue acts or natural language utterances (when paired with NLU and NLG). For most goal-oriented dialogues, the reward signal can be obtained from the user simulator or real user ratings. However, when that reward is not available, an output prediction model is required which takes as input the whole dialogue session

---

**Algorithm 1** OPPA for Dialogue Policy Learning

---

**Require:**  $\epsilon, C$ 

```

1: initialize  $\pi^o(s, a; \theta_\pi)$  and  $Q(s, a; \theta_Q)$  by su-
   pervised and imitation learning
2: initialize  $Q'(s, a, \theta_{Q'})$  with  $\theta_{Q'} = \theta_Q$ 
3: initialize replay buffer  $D$ 
4: for each iteration do
5:   user acts  $a^u$ 
6:   initialize state  $s$ 
7:   while not done do
8:      $e = \text{random}(0, 1)$ 
9:     if  $e < \epsilon$  then
10:      select a random action  $a$ 
11:    else
12:      sample  $\hat{a}$ 
13:      est. user action  $a_{t+1}^o = \pi^o(s, \hat{a}_t)$ 
14:       $\hat{s} = [s, E^o a_{t+1}^o]$ 
15:       $a = \text{argmax}_{a'} Q(\hat{s}, a'; \theta_Q)$ 
16:    end if
17:    execute  $a$ 
18:    get user response  $a^o$  and reward  $r$ 
19:    state updated to  $s'$ 
20:    store  $(s, a, r, s')$  to  $D$ 
21:  end while
22:  sample minibatches of  $(s, a, r, s')$  from  $D$ 
23:  update  $\theta_Q$  according to Equation 4
24:  each every  $C$  iterations set  $\theta_{Q'} = \theta_Q$ 
25: end for

```

---

$X = \{x_1^s, x_2^s, \dots, x_n^s\}$  where  $X$  is a sequence of tokens, and outputs structured result to calculate the reward.

We use a bi-directional GRU model with an attention mechanism to learn a summarization  $h^s$  of the whole session:

$$h_j^o = \text{BiGRU}(h_{j-1}^o, [Ex_j^s, h_j]), \quad (8)$$

$$h_j^a = W^a[\tanh(W^h h_j^o)], \quad (9)$$

$$\alpha_j = \frac{\exp(w \cdot h_j^a)}{\sum_{t'} \exp(w \cdot h_{t'}^a)}, \quad (10)$$

$$h^s = \tanh(W^s[h^g, \sum_j \alpha_j h_j]). \quad (11)$$

Note that in this process, we concatenated all the utterances by time order, and the subscript  $j$  indicates the index of word in the concatenated sequence. In addition, there may be multiple aspects of the output. For example, in a negotiation goal-oriented dialogue with multiple issues (we denote the book or hat items to negotiate on as issues), we

need to get the output of each issue to calculate the total reward. Therefore, for each issue  $o_i$ , a specific softmax classifier is applied:

$$p_\theta(o_i | x_{0 \dots T}, g) = \text{softmax}(W^{o_i} h^s). \quad (12)$$

After the structured output is predicted, we can obtain the final reward by applying the task-specific reward function on the output.

$$r = f^R(o_1, o_2, \dots, o_{N_o}), \quad (13)$$

where  $N_o$  is the number of output aspects and  $f^R$  is the reward function which is often manually defined according to the task.

## 4 Experiment

Depending on the task, dialogues can be divided into cooperative and competitive ones. In a cooperative task, the aim can be reducing unnecessary interactions by inferring the opposite person's intention. While in competitive tasks, the aim is usually to maximize their own interests by considering the opposite agents' possible reactions. To test our method's wide suitability, we evaluated it on both cooperative and competitive tasks.

### 4.1 Dataset

For the cooperative task, we used MultiWOZ ([Budzianowski et al., 2018](#)), a large-scale linguistically rich multi-domain goal-oriented dialogue dataset, which contains 7 domains, 13 intents and 25 slot types. There are 10,483 sessions and 71,544 turns, which is at least one order of magnitude larger than previous annotated task-oriented dialogue dataset. Among all the dialogue sessions, we used 1,000 each for validation and test. Specifically, in the data collection stage, the user follows a specific goal to converse with the agent but is encouraged to change his/her goal dynamically during the session, which makes the dataset more challenging.

For the competitive task, we used a bilateral negotiation dataset ([Lewis et al., 2017](#)), where there are 5,808 dialogues from 2,236 scenarios. In each session, there are two people negotiating to divide some items, such as books, hats and balls. Each kind of item is of different value to each person, thus they can give priority to valuable items in the negotiation. For example, a hat may worth 5 for person  $A$  and 3 for person  $B$ , so  $B$  can give up some hat in order to get other valuable items. To conduct our experiment, we further labeled the dataset with system dialogue actions.

## 4.2 Experimental Settings

We implemented the model using PyTorch (Paszke et al., 2017). The hyper-parameters were decided using validation set. The dimension of  $\text{GRU}_o$  hidden state is 256, and the hidden state size of  $\text{GRU}_g$  and  $\text{GRU}_w$  are 64 and 128 respectively. The size of  $h_s$  is 256. As for the Q-function, the size of  $s_t$  is 256.  $\epsilon$ -greedy is applied for exploration. The buffer size of  $D$  is set to 500 and the update step  $C$  is 1.

Note that due to the complexity of MultiWOZ, the error propagation problem caused by NLU and NLG is serious. Therefore, the cooperative experiment is conducted on the dialogue act level. In the experiment, our proposed model interacts with a robust rule-based user simulator, which appends an agenda-based model (Schatzmann et al., 2007) with extensive manual rules. The simulator gives user response, termination signal and goal-completion feedback during training. For the competitive task, the experiment is on natural language level. Following (Lewis et al., 2017), we built a seq2seq language model for the NLU and NLG module, which is pre-trained on the negotiation corpus.

Our proposed model was first pre-trained with supervised learning (SL). Specifically, we pre-trained the opposite estimator  $\pi_o$  and the Q-function  $Q(s, a)$  via supervised learning and imitation learning. We then fine-tuned the model using reinforcement learning (RL). The reward of the MultiWOZ experiment consists of two parts: a) a small negative value in each turn to encourage shorter sessions and b) a large positive reward when the session ends successfully. Note that the task completion signal is obtained from the user. For the negotiation experiment, the reward is the total value of item items that the agent finally got. In the negotiation dataset, the reward is given by the proposed output model described in the Reward Function section.

## 4.3 Baselines

To demonstrate the effectiveness of our proposed model, we compared it with the following baselines. For the MultiWOZ task, we compared with:

- **DQN**: The conventional DQN (Mnih et al., 2015) algorithm with a 2-layer fully-connected network for Q-function.
- **REINFORCE**: The REINFORCE algorithm (Williams, 1992) with a 2-layer fully-connected policy network.

- **PPO**: Proximal Policy Optimization (Schulman et al., 2017), a policy-based RL algorithm using a constant clipping mechanism.
- **DDQ**: The Deep Dyna-Q (Peng et al., 2018) algorithm which introduced a world-model for RL planning.

Note that the DQN can be seen as our proposed model without opposite estimator (**OPPA w/o OBE**). For the negotiation task, we compared with:

- **SL RNN**: A supervised learning method that is based on an RNN language generation model.
- **RL RNN**: The reinforcement learning extension of SL RNN by refining the model parameters after SL pretraining.
- **ROL**: SL RNN with goal-based decoding in which the model first generates several candidate utterances and chooses the one with the highest expected overall reward after rolling out several sessions.
- **RL ROL**: The extension of RL RNN with roll-out decoding.
- **HTG**: A hierarchical text generation model with planning (Yarats and Lewis, 2018), which learns explicit turn-level representation before generating a natural language response.

Note that the rollout mechanism used in ROL and RL ROL also endows them with the ability of “seeing ahead” in which the candidate actions’ rewards are predicted using a random search algorithm, while our OPPA explicitly models the opposite’s behavior. RL RNN, RL ROL and HTG used the REINFORCE (Williams, 1992) algorithm for reinforcement learning on both strategy and language level, while in OPPA we used the DQN (Mnih et al., 2015) algorithm only on strategy level. To further examine the effectiveness of our proposed action regularization with decay, we did an ablation study by removing the regularization with decay part in Equation 6 (**OPPA w/o A**).

## 4.4 Evaluation Metric

For the evaluation of experiments on MultiWOZ, we used the number of turns, inform F1 score, match rate and success rate. The **Number of turns** is the averaged number on all sessions, and less turns in cooperative goal-oriented task can promote user satisfaction. **Inform F1** evaluates whether all the slots of an entity requested by the user have been successfully informed. We use F1 score because it considers both the precision and recall so

that a policy which greedily informs all slot information of an entity won't get a high score. **Match rate** evaluates whether the booked entities match the goals in all domains. The score of a domain is 1 only when its entity is successfully booked. Finally, a session is considered **successful** only if all the requested slots are informed (recall = 1) and all entities are correctly booked.

For the negotiation task, we used the averaged scores (total values of items) of all the sessions and those with an agreement as the primary evaluation metrics following Lewis et al. (2017). The percentage of agreed and Pareto optimal\* sessions are also reported.

Method	#Turn	Inform F1	Match	Success
DQN	10.50	78.23	60.31	51.7
REINFORCE	9.49	81.73	67.41	58.1
PPO	9.83	83.34	69.09	59.0
DDQ	9.31	81.49	63.10	62.7
OPPA w/o A	8.19	88.45	77.18	75.2
OPPA	8.47	91.68	79.62	81.6
Human	7.37	66.89	95.29	75.0

Table 1: The results on MultiWoZ dataset, a large scale multi-domain task-oriented dialog dataset. We used a rule-based method for DST and Agenda-based user simulator. The DQN method can be regard as OPPA w/o OBE. Human-human performance from the test set serves as the upper bound.

#### 4.5 Cooperative Dialogue Analysis

The results on MultiWOZ dataset are shown in Table 1. OPPA shows superior performance on task success rate than other baseline methods due to the considerable improvement in Inform F1 and Match rate. By first infer the next action of the opposite agent, the target agent policy can make better choices to match the reward signal during training. When compared with human performance, OPPA even achieves a higher success rate, although the number of turns is still higher. This might be due to the fact that the user is sensitive to the dialogue length. When a dialogue becomes intolerably long, many user will leave without completing the dialogue. By taking actions in account of the inferred opposite action, the target agent can also make the dialogue more efficiently by avoiding some lengthy interactions, which is extremely important in applications where the user is sensitive to dialogue length.

Meanwhile, DDQ achieves higher task success rate than other baseline models since it also mod-

els the behavior of opposite agent through world model. However, it makes use of the learned world model by providing more simulated experiences, which does not give a direct hint on how to act in the middle of a session. Therefore, in its experiments, it still gets longer dialogue sessions and a lower success rate than OPPA.

If we remove the action regularization mechanism, we can see an obvious decline on performance, which is as expected. The action regularization is introduced to mitigate the difference between sampled  $\hat{a}_t$  and real  $a_t$ , so there can be a large discrepancy between the sampled and real actions if we remove it at the early training stage. When the  $\hat{a}_t$  is not reliable, the consequent estimated opposite action  $a'_{t+1}$  also becomes noisy, which leads to performance drop.

#### 4.6 Competitive Dialogue Analysis

Table 2 shows the scores for all sessions and for only agreed ones. When comparing with the seq2seq models, OPPA achieves significantly better results. This can be attributed to the hierarchical structure of OPPA. The sequence models only take as input (and outputs) the word-level natural language utterances, without explicitly modeling turn-level dialogue actions. In this way, the parameters for linguistic and strategy functions are tangled together, and the back-propagation errors can influence both sides. As for the two ROL models, although they can predict the value of a candidate action in advance, they still cannot beat OPPA. The reason is that the rollout method did not explicitly maintain an estimation of the opposite agent as our OPPA did. Instead, it just estimates the candidate actions' rewards based on Monte Carlo search by using its own model for predicting future movements. Therefore, when the opposite model's behavior is not very familiar to the target agent, the estimated reward becomes unreliable.

The HTG model also used a hierarchical framework by learning an explicit turn-level latent representation. By doing this, it obtains higher scores than the seq2seq models. However, it does not make any assumptions about the opposite agent. Therefore, its scores are still lower than OPPA, although the discrepancy narrows down.

By removing the opposite estimator, we find that the performance of OPPA w/o OBE drops significantly compared to that of OPPA. This ablation study directly verifies the effectiveness of our proposed opposite behavior estimator. There fore,

\* A dialogue is Pareto optimal if neither agent's score can be improved without lowering the other's score.

Method	vs. SL RNN		vs. RL RNN		vs. ROL		vs. RL ROL	
	All	Agreed	All	Agreed	All	Agreed	All	Agreed
SL RNN	5.4 vs. 5.5	6.2 vs. 6.2	-	-	-	-	-	-
RL RNN	7.1 vs. 4.2	7.9 vs. 4.7	5.5 vs. 5.6	5.9 vs. 5.8	-	-	-	-
ROL	7.3 vs. 5.1	7.9 vs. 5.5	5.7 vs. 5.2	6.2 vs. 5.6	5.5 vs. 5.4	5.8 vs. 5.9	-	-
RL ROL	8.3 vs. 4.2	8.8 vs. 4.5	5.8 vs. 5.0	6.5 vs. 5.5	6.2 vs. 4.9	7.0 vs. 5.4	5.9 vs. 5.8	6.4 vs. 6.3
HTG	8.7 vs. 4.4	8.8 vs. 4.5	6.0 vs. 5.1	6.9 vs. 5.5	6.5 vs. 5.0	6.9 vs. 5.3	6.5 vs. 5.6	7.0 vs. 6.3
OPPA w/o OE	8.2 vs. 4.2	8.8 vs. 4.7	6.1 vs. 5.2	6.8 vs. 5.6	6.5 vs. 4.8	7.0 vs. 5.3	5.7 vs. 5.8	6.5 vs. 6.4
OPPA w/o A	8.7 vs. 4.1	8.9 vs. 4.3	6.3 vs. 5.0	7.2 vs. 5.4	6.5 vs. 4.8	7.2 vs. 5.4	6.5 vs. 6.1	7.1 vs. 6.8
OPPA	8.8 vs. 3.9	9.0 vs. 4.1	6.7 vs. 4.6	7.3 vs. 5.2	6.8 vs. 4.2	7.4 vs. 5.1	6.7 vs. 6.0	7.2 vs. 6.6

Table 2: The results of our proposed OPPA and the baselines on the negotiation dataset. *All* and *Agreed* indicates averaged scores for all sessions and only the agreed sessions respectively.

Method	vs. SL RNN		vs. RL RNN		vs. ROL		vs. RL ROL	
	Agreed(%)	PO(%)	Agreed(%)	PO(%)	Agreed(%)	PO(%)	Agreed(%)	PO(%)
SL RNN	87.9	49.6	-	-	-	-	-	-
RL RNN	89.9	58.6	81.5	60.3	-	-	-	-
ROL	92.9	63.7	87.4	65.0	85.1	67.3	-	-
RL ROL	94.4	74.8	85.7	74.6	71.2	76.4	67.5	77.2
HTG	94.8	75.1	88.3	75.4	83.2	77.8	66.1	73.2
OPPA w/o OBE	94.6	74.6	87.9	75.2	79.3	78.2	73.7	77.9
OPPA w/o A	95.6	77.9	91.9	77.4	82.4	78.8	78.0	79.5
OPPA	95.7	77.7	91.4	77.2	82.3	79.1	78.2	79.7

Table 3: The proportion of agreed and Pareto optimal (PO) sessions for our proposed OPPA and the baselines on the negotiation dataset.

modeling the opposite policy in one’s mind is a crucial source to achieve better results in competitive dialogue policy learning.

When comparing with OPPA w/o A which removed action regularization, we can see that the OPPA model gets better results. This verifies the importance of regularizing the action sampling. By controlling the difference between real and model generated actions, we can keep the opposite model consistent with the real opposite agent at the early training stage.

The percentage of agreed and Pareto optimal session are shown in Table 3. As we can see, the percentage of Pareto optimal increases in our method, showing that the OPPA model can explore the solution space more effectively. However, the agreement rate decreases when the opposite model gets stronger. This phenomenon is also found in Lewis et al. (2017) when they change the opposite agent from SL RNN to real human. This can be attributed to the aggressiveness of the agent: when both agents act aggressively, they are less likely to reach an agreement. The SL RNN model simply imitates the behavior in the dialogue corpus, while the ROL and RL mechanisms both help the agent to explore more spaces, which makes them more aggressive on action selection.

#### 4.7 Human Evaluation

To better validate our propositions, we further conducted human evaluation by making our model conversing with real user. We only conducted human evaluation on the negotiation task since the MultiWOZ model is implemented on the dialogue act level. We tested the models on a total of 1,000 dialogue sessions. In the evaluation, the users conversed with the agent, and the total item values are used as the evaluation metrics. The results are shown in Table 4. We can see that our proposed OPPA outperforms the baseline models. The system score are lower than that in Table 2, and the discrepancy between *All* and *Agreed* results is large. This can be due to the high intelligence and aggressiveness of real humans who want to get as more values as possible and do not make compromises easily. Due to this reason, the sessions become considerably lengthy, and the target agent exceeds our length limit before reaching an agreement.

Method	All	Agreed
RL ROL	4.5 vs. 5.2	7.8 vs. 7.1
HTG	4.8 vs. 4.7	8.0 vs. 7.2
OPPA w/o A	4.7 vs. 4.9	8.4 vs. 6.7
OPPA	5.2 vs. 5.1	8.2 vs. 6.5

Table 4: The rewards of each model vs. human user.

#### 5 Conclusion

In this work, we present an opposite agent-aware dialogue policy model which actively estimates the

opposite agent instead of doing passive learning from experiences. We have shown that it is possible to harvest a reliable model of the opposite agent through more efficient dialogue interactions. By incorporating the estimated model output as part of dialogue state, the target agent shows significant improvement on both cooperative and competitive goal-oriented tasks. As future work, we will explore multi-party dialogue modeling in which multi-agent learning techniques can be applied.

## Acknowledgments

This work was jointly supported by the NSFC projects (key project with No. 61936010 and regular project with No. 61876096), and the Guoqiang Institute of Tsinghua University with Grant No. 2019GQG1. This work was also supported by Beijing Academy of Artificial Intelligence, BAAI and Beijing Nova Program (Z201100006820068) from Beijing Municipal Science & Technology Commission. We thank THUNUS NExT Joint-Lab for the support.

## References

- Layla El Asri, Jing He, and Kaheer Suleman. 2016. A sequence-to-sequence model for user simulation in spoken dialogue systems. *arXiv preprint arXiv:1607.00070*.
- Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Inigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. Multiwoz-a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. *arXiv preprint arXiv:1810.00278*.
- Inigo Casanueva, Paweł Budzianowski, Pei-Hao Su, Stefan Ultes, Lina Rojas-Barahona, Bo-Hsiang Tseng, and Milica Gašić. 2018. Feudal reinforcement learning for dialogue management in large domains. *arXiv preprint arXiv:1803.03232*.
- Heriberto Cuayahuitl, Seunghak Yu, Ashley Williamson, and Jacob Carse. 2016. Deep reinforcement learning for multi-domain dialogue systems. *arXiv preprint arXiv:1611.08675*.
- Bhuwan Dhingra, Lihong Li, Xiujun Li, Jianfeng Gao, Yun-Nung Chen, Faisal Ahmed, and Li Deng. 2016. Towards end-to-end reinforcement learning of dialogue agents for information access. *arXiv preprint arXiv:1609.00777*.
- Mehdi Fatemi, Layla El Asri, Hannes Schulz, Jing He, and Kaheer Suleman. 2016. Policy networks with two-stage training for dialogue systems. *arXiv preprint arXiv:1606.03152*.
- Vittorio Gallese and Alvin Goldman. 1998. Mirror neurons and the simulation theory of mind-reading. *Trends in cognitive sciences*, 2:493–501.
- Jianfeng Gao, Michel Galley, Lihong Li, et al. 2019. Neural approaches to conversational ai. *Foundations and Trends® in Information Retrieval*, 13(2-3):127–298.
- M Gašić, N Mrkšić, Pei-hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2015. Policy committee for adaptation in multi-domain spoken dialogue systems. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 806–812. IEEE.
- Robert M Gordon. 1986. Folk psychology as simulation. *Mind & Language*, 1(2):158–171.
- Izzeddin Gür, Dilek Hakkani-Tür, Gokhan Tür, and Pararth Shah. 2018. User modeling for task oriented dialogues. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 900–906. IEEE.
- Mary Harper. 2014. Learning from 26 languages: Program management and science in the babel program. In *COLING*, page 1.
- Megha Jhunjhunwala, Caleb Bryant, and Pararth Shah. 2020. Multi-action dialog policy learning with interactive human teaching. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 290–296, 1st virtual meeting. Association for Computational Linguistics.
- Florian Kreyssig, Inigo Casanueva, Paweł Budzianowski, and Milica Gašić. 2018. Neural user simulation for corpus-based policy optimisation for spoken dialogue systems. *arXiv preprint arXiv:1805.06966*.
- Oliver Lemon and Olivier Pietquin. 2012. *Data-driven methods for adaptive spoken dialogue systems: Computational learning for conversational interfaces*. Springer Science & Business Media.
- Esther Levin, Roberto Pieraccini, and Wieland Eckert. 1997. Learning dialogue strategies within the markov decision process framework. In *1997 IEEE Workshop on Automatic Speech Recognition and Understanding Proceedings*, pages 72–79. IEEE.
- Mike Lewis, Denis Yarats, Yann Dauphin, Devi Parikh, and Dhruv Batra. 2017. Deal or no deal? end-to-end learning of negotiation dialogues. In *EMNLP*.
- Xiujun Li, Yun-Nung Chen, Lihong Li, Jianfeng Gao, and Asli Celikyilmaz. 2017. End-to-end task-completion neural dialogue systems. *arXiv preprint arXiv:1703.01008*.
- Xiujun Li, Zachary C Lipton, Bhuwan Dhingra, Lihong Li, Jianfeng Gao, and Yun-Nung Chen. 2016. A user simulator for task-completion dialogues. *arXiv preprint arXiv:1612.05688*.

- Zachary Lipton, Xiujun Li, Jianfeng Gao, Lihong Li, Faisal Ahmed, and Li Deng. 2018. Bbq-networks: Efficient exploration in deep reinforcement learning for task-oriented dialogue systems. In *AAAI*.
- Bing Liu and Ian Lane. 2017. Iterative policy learning in end-to-end trainable task-oriented neural dialog models. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 482–489. IEEE.
- Bing Liu, Gokhan Tur, Dilek Hakkani-Tur, Pararth Shah, and Larry Heck. 2018. Dialogue learning with human teaching and feedback in end-to-end trainable task-oriented dialogue systems. *arXiv preprint arXiv:1804.06512*.
- Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529.
- Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch. In *NIPS 2017 Workshop Autodiff*.
- Baolin Peng, Xiujun Li, Jianfeng Gao, Jingjing Liu, and Kam-Fai Wong. 2018. Deep dyna-q: Integrating planning for task-completion dialogue policy learning. In *ACL*.
- Baolin Peng, Xiujun Li, Lihong Li, Jianfeng Gao, Asli Celikyilmaz, Sungjin Lee, and Kam-Fai Wong. 2017. Composite task-completion dialogue policy learning via hierarchical deep reinforcement learning. In *EMNLP*.
- David Premack and Guy Woodruff. 1978. Does the chimpanzee have a theory of mind? *Behavioral Linguistics*, page 1.
- Verena Rieser and Oliver Lemon. 2011. *Reinforcement learning for adaptive dialogue systems: a data-driven methodology for dialogue management and natural language generation*. Springer Science & Business Media.
- Jost Schatzmann, Blaise Thomson, Karl Weilhammer, Hui Ye, and Steve Young. 2007. Agenda-based user simulation for bootstrapping a pomdp dialogue system. In *NAACL*, pages 149–152.
- Jost Schatzmann, Karl Weilhammer, Matt Stuttle, and Steve Young. 2006. A survey of statistical user simulation techniques for reinforcement-learning of dialogue management strategies. *The knowledge engineering review*, 21(2):97–126.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.
- David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. 2016. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484.
- Pei-Hao Su, Milica Gasic, Nikola Mrksic, Lina Rojas-Barahona, Stefan Ultes, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016. Continuously learning neural dialogue management. *arXiv preprint arXiv:1606.02689*.
- Shang-Yu Su, Xiujun Li, Jianfeng Gao, Jingjing Liu, and Yun-Nung Chen. 2018. Discriminative deep dyna-q: Robust planning for dialogue policy learning. In *EMNLP*.
- Richard S Sutton. 1990. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Machine Learning Proceedings 1990*, pages 216–224. Elsevier.
- Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *NIPS*.
- Ryuichi Takanobu, Runze Liang, and Minlie Huang. 2020. Multi-agent task-oriented dialog policy learning with role-aware reward decomposition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 625–638, Online. Association for Computational Linguistics.
- Ryuichi Takanobu, Hanlin Zhu, and Minlie Huang. 2019. Guided dialog policy learning: Reward estimation for multi-domain task-oriented dialog. *arXiv preprint arXiv:1908.10719*.
- Zhuoran Wang, Hongliang Chen, Guanchun Wang, Hao Tian, Hua Wu, and Haifeng Wang. 2014. Policy learning for domain selection in an extensible multi-domain spoken dialogue system. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 57–67.
- Théophane Weber, Sébastien Racanière, David P Reichert, Lars Buesing, Arthur Guez, Danilo Jimenez Rezende, Adria Puigdomenech Badia, Oriol Vinyals, Nicolas Heess, Yujia Li, et al. 2017. Imagination-augmented agents for deep reinforcement learning. *arXiv preprint arXiv:1707.06203*.
- Jason D Williams, Kavosh Asadi, and Geoffrey Zweig. 2017. Hybrid code networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning. *arXiv preprint arXiv:1702.03274*.
- Jason D Williams and Steve Young. 2007. Partially observable markov decision processes for spoken dialog systems. *Computer Speech & Language*, 21:393–422.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256.

Yuxin Wu, Xiujun Li, Jingjing Liu, Jianfeng Gao, and Yiming Yang. 2018. Switch-based active deep dyna-q: Efficient adaptive planning for task-completion dialogue policy learning. *arXiv preprint arXiv:1811.07550*.

Denis Yarats and Mike Lewis. 2018. Hierarchical text generation and planning for strategic dialogue. In *ICML*, pages 5587–5595.

Steve Young, Milica Gašić, Blaise Thomson, and Jason D Williams. 2013. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179.

Wen Zhang, Yang Feng, Fandong Meng, Di You, and Qun Liu. 2019a. Bridging the gap between training and inference for neural machine translation. In *ACL*, pages 4334–4343.

Zhirui Zhang, Xiujun Li, Jianfeng Gao, and Enhong Chen. 2019b. Budgeted policy learning for task-oriented dialogue systems. *arXiv preprint arXiv:1906.00499*.

Tiancheng Zhao and Maxine Eskenazi. 2016. Towards end-to-end learning for dialog state tracking and management using deep reinforcement learning. *arXiv preprint arXiv:1606.02560*.

# An Empirical Study of Tokenization Strategies for Various Korean NLP Tasks

Kyubyong Park<sup>1,\*</sup> Joohong Lee<sup>2,\*</sup> Seongbo Jang<sup>2,3,\*</sup> Dawoon Jung<sup>2,\*</sup>

<sup>1</sup>Kakao Brain, <sup>2</sup>Pingpong AI Research, Scatter Lab,

<sup>3</sup>Pohang University of Science and Technology

kyubyong.park@kakaobrain.com

{jooohong,dawoon}@scatterlab.co.kr

jang.sb@postech.ac.kr

## Abstract

Typically, tokenization is the very first step in most text processing works. As a token serves as an atomic unit that embeds the contextual information of text, how to define a token plays a decisive role in the performance of a model.

Even though Byte Pair Encoding (BPE) has been considered the *de facto* standard tokenization method due to its simplicity and universality, it still remains unclear whether BPE works best across all languages and tasks. In this paper, we test several tokenization strategies in order to answer our primary research question, that is, “What is the best tokenization strategy for Korean NLP tasks?”

Experimental results demonstrate that a hybrid approach of morphological segmentation followed by BPE works best in Korean to/from English machine translation and natural language understanding tasks such as KorNLI, KorSTS, NSMC, and PAWS-X. As an exception, for KorQuAD, the Korean extension of SQuAD, BPE segmentation turns out to be the most effective.

Our code and pre-trained models are publicly available at <https://github.com/kakaobrain/kortok>.

## 1 Introduction

Tokenization is the very first step in most text processing works. Not surprisingly, tremendous academic efforts have been made to find the best tokenization method for various NLP tasks. For the past few years, Byte Pair Encoding (BPE) (Gage, 1994) has been considered the *de facto* standard tokenization technique since it was reintroduced by Sennrich et al. (2016a). Besides the fact that BPE turns out to be very effective in the machine translation task, another important reason BPE has gained

such popularity is that BPE is a data-driven statistical algorithm so it is independent of language. However, it is still not clear whether BPE works best across all languages, irrespective of tasks.

In this paper we study various tokenization strategies for Korean, a language which is morphologically by far richer than English. Concretely, we empirically examine what is the best tokenization strategy for Korean to English / English to Korean machine translation tasks, and natural language understanding (NLU) tasks—machine reading comprehension (MRC), natural language inference (NLI), semantic textual similarity (STS), sentiment analysis, and paraphrase identification. We are particularly interested in how complementary BPE and linguistically motivated segmentation are.

## 2 Background

### 2.1 MeCab-ko: A Korean Morphological Analyzer

MeCab (Kudo, 2006) is an open-source morphological analyzer based on Conditional Random Fields (CRFs). It is originally designed for Japanese, but also serves generic purposes so it can be applied to other languages. MeCab-ko<sup>1</sup>, a Korean extension of MeCab, started from the idea that MeCab can be easily extended to the Korean language due to the close similarity between Japanese and Korean in terms of morphology or syntax.

MeCab-ko trained its model on the Sejong Corpus (Kang and Kim, 2001), arguably the largest Korean corpus morphologically annotated by many experts, using MeCab. Ever since released in 2013, MeCab-ko has been widely used for many Korean NLP tasks due to its high accuracy and good usability. For example, the Workshop on Asian Transla-

\*Equal contribution.

<sup>1</sup><https://bitbucket.org/eunjeon/mecab-ko>

Tokenization	Tokenized Sequence
Raw Text	나랑 쇼핑하자.
CV (4.1)	ㄴ / ㅏ / 랑 / ㄱ / ㅗ / ㅓ / ㅍ / ㅎ / ㅏ / ㅈ / ㅏ / .
Syllable (4.2)	나/랑/*/쇼/핑/하/자/.
Morpheme (4.3)	나/랑/*/쇼/핑/하/자/.
Subword (4.4)	_나랑/_쇼/_핑_하/_자/_.
Morpheme-aware Subword (4.5)	_나/_랑/*/_쇼/_핑/_하/_자/_.
Word (4.6)	나랑/쇼/핑/하/자/.

Table 1: An input sentence 나랑 쇼핑하자. ‘Let’s go shopping with me.’ is differently tokenized depending on the various tokenization strategies. Slashes (/) are token separators.

tion (WAT) has adopted it as the official segmentation tool for evaluating Korean machine translation results since 2015. (Nakazawa et al., 2015, 2016, 2017, 2018, 2019).

## 2.2 Byte Pair Encoding

Byte Pair Encoding (BPE) is a simple data compression technique that iteratively replaces the most frequent pair of bytes in text with a single, unused byte (Gage, 1994). Since Sennrich et al. (2016b) successfully applied it to neural machine translation models, it has been regarded as the standard tokenization method across languages.

Korean is not an exception; Park et al. (2019b) applied BPE to the Korean text in the Korean to Japanese task of WAT 2019 and ranked first. In addition, most recent Korean neural language models (e.g., KoBERT<sup>2</sup>) used BPE to tokenize the training text.

## 3 Related Work

There have been extensive studies about tokenization techniques for machine translation. Several papers claimed that a hybrid of linguistically informed segmentation and a data-driven method like BPE or unigram language modeling performs the best for non-English languages. Banerjee and Bhattacharyya (2018) combined an off-the-shelf morphological segmenter and BPE in Hindi and Bengali translations against English. Tawfik et al. (2019) used a retrained version of linguistically motivated segmentation model along with statistical segmentation methods for Arabic. Pinnis et al. (2017) adopted linguistic guidance to BPE for English-Latvian translation. Particularly (Park et al., 2019a) is close to ours, but their main focus is on preprocessing techniques for neural machine

translation like parallel corpus filtering rather than on tokenization strategies per se.

Compared with the tokenization studies for machine translation, those for NLU tasks have gained less attention. Among them is Bostrom and Durrett (2020), which compared the fine-tuning task performance of BERT (Devlin et al., 2019) pre-trained with BPE and unigram language modeling. Moon and Okazaki (2020) proposed a novel encoding method for Korean and showed its efficiency in vocabulary compression with a few Korean NLU datasets.

## 4 Tokenization Strategies

We introduce assorted Korean tokenization strategies arranged from the smallest to the largest unit. Each of them induces different tokenization results, as illustrated in Table 1.

### 4.1 Consonant and Vowel (CV)

In Hangul, the standard Korean writing system, consonants and vowels, called *Jamo* in Korean, corresponding to Latin letters are assembled to form a syllable character. For example, a Hangul consonant ㅋ /h/ (U+314E) is combined with a vowel ㅏ /a/ (U+314F) to make a syllable character ㅋㅏ /ha/ (U+558). Readers who are not familiar with such a mechanism can think of Jamo and syllables as atoms and molecules respectively. As a molecule H<sub>2</sub>O can be decomposed into two H atoms and an O atom, a syllable ㅋㅏ /ha/ can be decomposed into its constituent consonant ㅋ /h/ and vowel ㅏ /a/. The first syllable ㅓ /na/ of the raw text in Table 1 is tokenized into ㅓ /n/ and ㅏ /a/, and the second syllable ㅏ랑 /lang/ is tokenized into ㅏ /l/, ㅏ /a/, and ㅇ /ng/, and so on. A whitespace is replaced by a special symbol \*.

<sup>2</sup><https://github.com/SKTBrain/KoBERT>

## 4.2 Syllable

We can tokenize a sentence at the syllable level. A whitespace is replaced by the special symbol  $\star$ .

## 4.3 Morpheme

MeCab-ko provides a convenient tokenization option in the command line interface<sup>3</sup>. For example, it returns A, B, and C given an input text AB C, where A-C represent morphemes. Note that the original space between AB and C is missing in the output token list. Accordingly, it is NOT possible to recover the original text from the tokenized result.

This can be problematic in some tasks that require us to restore the input text such as machine translation whose target language is Korean, or machine reading comprehension where we are expected to suggest a certain phrase in the given text as the answer.

For this reason, we insert a special token  $\star$  (U+2B51) to the original whitespace position. As a result, in the above example, the tokenized sequence looks like A, B,  $\star$ , and D.

## 4.4 Subword

We learn and apply BPE using the SentencePiece ([Kudo and Richardson, 2018](#)) library. It prepends ‘\_’ (U+2581) to every word to mark the original whitespace, then tokenizes text into subword pieces. As seen in Table 1, 나랑 쇼핑하자. can be split into \_나랑, \_쇼, 평하, 자, and . (period).

## 4.5 Morpheme-aware Subword

Motivated by the combined methods of data- and linguistically-driven approaches ([Banerjee and Bhattacharyya, 2018; Park et al., 2019a; Pinnis et al., 2017; Tawfik et al., 2019](#)), we apply MeCab-ko and BPE in sequence to make morpheme-aware subwords. According to this strategy, since BPE is applied *after* the original text is split into morphemes, tokens spanning multiple morphemes (e.g., 평하 in the Section 4.4) are not generated. Instead, the BPE algorithm further segments morphemes into frequent pieces.

## 4.6 Word

We can simply split text by whitespaces. Note that punctuation marks are split into separate tokens. Check that 나랑 쇼핑하자. is tokenized into 나랑, 쇼핑하자 and . (period) in Table 1.

<sup>3</sup>% mecab -O wakati

Lang Pair	Vocab Size	Korean BPE Training Data	Dev	Test
Ko-En	32K	AI Hub (130MB)	35.79	36.06
		Wiki (613MB)	<b>39.05</b>	<b>38.69</b>
En-Ko	32K	AI Hub (130MB)	<b>37.19</b>	<b>36.98</b>
		Wiki (613MB)	37.11	<b>36.98</b>

Table 2: BLEU scores of Korean to English (Ko-En) and English to Korean (En-Ko) translation models with different BPE training data. Note that the English sentences are tokenized using a 32K BPE model trained on the English Wiki.

## 5 Experiments

### 5.1 Korean to/from English Machine Translation

#### 5.1.1 Dataset

To date, there have yet been few open source benchmark datasets for Korean-English machine translation, not to mention that Korean is not in the language list of WMT<sup>4</sup> or IWSLT<sup>5</sup>. [Park et al. \(2019a\)](#) used OpenSubtitles ([Lison and Tiedemann, 2016](#)), a collection of crowd-sourced movie subtitles across 65 different languages, for English to Korean translation, but they are too noisy to serve as a translation benchmark dataset.<sup>6</sup>

Recently, a Korean-English parallel corpus was publicly released by AI Hub<sup>7</sup>, which was gathered from various sources such as news, government web sites, legal documents, etc. We download the news data, which amount to 800K sentence pairs, and randomly split them into 784K (train), 8K (dev), and 8K (test).

#### 5.1.2 BPE Modeling

Prior to training, we do simple preliminary experiments to decide which dataset to use for learning BPE.

There are two choices: AI Hub news training data and open source large text such as Wiki. AI Hub training data is relatively small in size (130 MB), but can be optimal as its lexical distribution will be close to that of the test data, considering both of them are from the same source. On the other hand, Wiki is larger, but it is not news per se, so can be not as appropriate as AI Hub data for

<sup>4</sup><https://www.aclweb.org/anthology/venues/wmt>

<sup>5</sup><http://iwslt.org/doku.php?id=start>

<sup>6</sup>[Park et al. \(2019a\)](#) reported BLEU scores of 7-12.

<sup>7</sup><http://www.aihub.or.kr/aidata/87>

Tokenization	Vocab Size	Ko-En		En-Ko		OOV Rate (%)	Avg. Length
		Dev	Test	Dev	Test		
CV	166	39.11	38.56	36.52	36.45	0.02	142.75
Syllable	2K	39.30	38.75	38.64	38.45	0.06	69.20
Morpheme	8K	31.59	31.24	32.44	32.19	7.51	49.19
	16K	34.38	33.80	35.74	35.52	4.67	49.19
	32K	36.19	35.74	36.51	36.12	2.72	49.19
	64K	<u>37.88</u>	<u>37.37</u>	<u>37.51</u>	<u>37.03</u>	1.40	49.19
	4K	39.18	38.75	<u>38.31</u>	<u>38.18</u>	0.07	48.02
Subword	8K	39.16	38.75	38.09	37.94	0.08	38.44
	16K	<u>39.22</u>	<u>38.77</u>	37.64	37.34	0.10	33.69
	32K	39.05	38.69	37.11	36.98	0.11	30.21
	64K	37.02	36.46	35.77	35.64	0.12	27.50
	4K	39.41	38.95	39.29	39.13	0.06	65.17
Morpheme-aware Subword	8K	39.42	39.06	39.78	39.61	0.06	56.79
	16K	39.84	39.41	40.23	40.04	0.07	53.30
	32K	<b>41.00</b>	<b>40.34</b>	<b>40.43</b>	<b>40.41</b>	0.07	51.38
	64K	39.62	39.34	38.63	38.42	0.07	50.27
Word	64K	7.04	7.07	18.68	18.42	26.20	18.96

Table 3: BLEU scores of Korean to English (**Ko-En**) and English to Korean (**En-Ko**) translation models of various tokenization strategies. Note that we use an 32K Subword model for English for all of them. The OOV rate values in the table are obtained from the test set, but there is no meaningful difference between the test and the dev set in terms of the OOV rate. The best BLEU scores in each column (global) and group (local) are bold-faced and underlined, respectively.

BPE modeling.

To investigate this, first we train a 32K Korean BPE model (**A**) using SentencePiece with the Korean sentences in the AI Hub training data. Then we download the latest Wikipedia Korean<sup>8</sup>/English<sup>9</sup> dumps, and extract plain texts using WikiExtractor<sup>10</sup>. Next, we make 32K BPE models for Korean (**B**) and English (**C**) with them. Finally, we train Korean to English (Ko-En) and English to Korean (En-Ko) translation models on the AI Hub training data with the two different Korean BPE models (**A, B**). The training details are explained in Section 5.1.3. For comparison, we use the same English BPE model (**C**) for both.

The results are shown in Table 2. For Ko-En translation, the Wiki-based BPE model performs better in both dev and test sets by 2-3 points. For En-Ko translation, there is no practical difference in performance between the Wiki and AI Hub-based models. It is also worth considering the BPE models are used for NLU tasks as well as machine translation. All things taken together, we opt for

the Wiki-based BPE model.

### 5.1.3 Training

We test the tokenization strategies in Section 4 with various vocabulary sizes on the AI Hub news dataset.

We use the Transformer (Vaswani et al., 2017), the state-of-the-art model for neural machine translation. We mostly follow the base model configuration: 6 blocks of 512-2048 units with 8 attention heads. We run all of our experiments using FAIRSEQ<sup>11</sup> (Ott et al., 2019), a PyTorch based deep learning library for sequence to sequence models.

Each model is trained using a Tesla V100 GPU with batch size 128, dropout rate 0.3, label smoothing 0.1, and the Adam (Kingma and Ba, 2015) optimizer. We set the learning rate to 5e-4 with the inverse square-root schedule. We train all models for 50 epochs and save the checkpoint files at every epoch.

### 5.1.4 Results

After all training stages are finished, we evaluate the saved checkpoint files of each model on

<sup>8</sup><https://dumps.wikimedia.org/kowiki>

<sup>9</sup><https://dumps.wikimedia.org/enwiki>

<sup>10</sup><https://github.com/attardi/wikiextractor>

<sup>11</sup><https://github.com/pytorch/fairseq>

Vocab Size	# Tokens	# Tokens Spanning Morpheme Boundaries
4K	387,088	25,458 (6.58%)
8K	309,360	50,029 (16.17%)
16K	271,334	62,861 (23.17%)
32K	242,736	73,609 (30.26%)
64K	221,530	82,324 (37.16%)

Table 4: Number of tokens spanning morpheme boundaries in Subword models.

the dev set to find the best one, which is subsequently used for the final test. In Table 3 we report BLEU scores on both the dev and test sets using the Moses<sup>12</sup> multi-bleu.perl script. Following WAT 2019 (Nakazawa et al., 2019), Moses tokenizer and MeCab-ko are used for tokenizing the evaluation data.

For both Ko-En and En-Ko, overall, the Subword models (35.64-39.22) and the Syllable models (38.45-39.30) are superior to the Morpheme models (31.59-37.37) or the Word models (7.04-18.42) in performance. It is highly likely to come from the lower OOV rates of the Subword models (0.07-0.12) and the Syllable models (0.06) compared to those of the Morpheme models (1.40-7.51) and the Word models (26.20). While BPE tends to split rare words into subword pieces, MeCab-ko is ignorant of statistics so it splits words into morphemes by linguistic knowledge instead. That the Morpheme and Word models generate many OOVs suggests Korean has so large types of morphemes or word forms that even 64K vocabulary is not enough to cover them all.

CV models are tiny in vocabulary size (166) so they show the lowest OOV rate (0.02). However, their performance is not as good as the Syllable or Subword models. We speculate this is because a single consonant or vowel must bear too much contextual information in the CV models.

Morpheme-aware Subword 32K models achieve the best BLEU scores. Each Subword model, as shown in Table 4, contains 6-37% of tokens spanning morpheme boundaries in the test set, which implies that subword segmentation by BPE is not optimal and morpheme boundaries are meaningful in tokenization.

To sum up, morpheme-aware subword tokenization that makes the best use of linguistic knowledge and statistical information is the best for Korean machine translation.

<sup>12</sup><http://www.statmt.org/moses>

Hyper-param	KorQuAD	KorNLI	KorSTS	NSMC	PAWS
Epoch	5	3	5	3	5
Batch	16	64	64	64	64
$\eta$	5e-5	1e-4	5e-5	5e-5	1e-4
Dropout	0.1	0.1	0.1	0.1	0.1
Warm-up	0.1	0.1	0.1	0.1	0.1
Max Seq. <sup>†</sup>	128	128	128	128	128

Table 5: Fine-tuning hyper-parameters for NLU tasks.  $\eta$ : learning rate. <sup>†</sup>: Max sequence length is 256 for CV models in all tasks.

## 5.2 Korean Natural Language Understanding Tasks

Large pre-trained language models have proven their effectiveness in many downstream tasks (Peters et al., 2018; Devlin et al., 2019; Liu et al., 2019). We pre-train BERT (Devlin et al., 2019) models with various tokenization strategies, and fine-tune them on five different Korean NLU tasks.

### 5.2.1 Machine Reading Comprehension: KorQuAD 1.0 Dataset

The KorQuAD 1.0 dataset (Lim et al., 2019) is a Korean adaptation of SQuAD 1.0 (Rajpurkar et al., 2016), a popular reading comprehension dataset. KorQuAD 1.0 consists of 10,645 passages and their paired 66,181 questions (60,407 for training + 5,774 for development<sup>13</sup>). Like SQuAD 1.0, KorQuAD 1.0 involves answering a question given a passage. The answer must be a phrase within the passage.

### 5.2.2 Natural Language Inference: KorNLI Dataset

The KorNLI Dataset (Ham et al., 2020) is a Korean NLI dataset sourced from three different NLI datasets: SNLI (Bowman et al., 2015), MNLI (Williams et al., 2018), and XNLI (Conneau et al., 2018).

It is composed of 950,354 sentence pairs: 942,854 for training, 2,490 for development, and 5,010 for test. A model receives a pair of sentences—a premise and a hypothesis—and classifies their relationship into one out of three categories: *entailment*, *contradiction*, and *neutral*.

### 5.2.3 Semantic Textual Similarity: KorSTS Dataset

The KorSTS Dataset (Ham et al., 2020) is a Korean STS dataset translated from the STS-B dataset (Cer et al., 2017). It comprises 8,628 sentence

<sup>13</sup>The test dataset is not included.

Tokenization	Vocab	KorQuAD	KorNLI	KorSTS		NSMC		PAWS-X	
	Size	Dev (EM/F1)	Dev	Test	Dev	Test	Dev	Test	Dev
CV	166	59.66 / 73.91	70.60	71.20	77.22	71.47	87.97	87.89	58.00
Syllable	2K	69.10 / 83.29	73.98	73.47	82.70	75.86	88.94	89.07	68.65
Morpheme	32K	68.05 / 83.82	74.86	74.37	82.37	76.83	87.87	88.04	69.30
	64K	<u>70.68</u> / <u>85.25</u>	<u>75.06</u>	<u>75.69</u>	<u>83.21</u>	<u>77.38</u>	<u>88.72</u>	<u>88.88</u>	<u>73.40</u>
Subword	4K	71.48 / 83.11	74.38	74.03	83.37	76.80	89.08	89.30	72.00
	8K	72.91 / 85.11	74.18	74.65	83.23	76.42	89.08	89.19	73.45
	16K	73.42 / 85.75	74.46	<u>75.15</u>	83.30	76.41	88.89	88.88	73.40
	32K	<b>74.04</b> / 86.30	<u>74.74</u>	74.29	83.02	77.01	<u>89.39</u>	<u>89.38</u>	74.05
	64K	<b>74.04</b> / <b>86.66</b>	73.73	74.55	<u>83.52</u>	<u>77.47</u>	88.80	89.19	<u>75.85</u>
Morpheme-aware Subword	4K	67.53 / 81.93	73.53	73.45	83.34	76.03	88.93	89.32	69.75
	8K	70.90 / 84.57	74.14	73.95	83.71	76.07	89.37	89.29	73.40
	16K	69.47 / 83.36	75.02	74.99	83.22	76.59	89.33	89.41	75.05
	32K	<u>72.65</u> / <u>86.35</u>	74.10	75.13	83.65	<b>78.11</b>	89.53	89.65	74.60
	64K	69.48 / 83.73	<b>76.39</b>	<b>76.61</b>	<b>84.29</b>	76.78	<b>89.82</b>	<b>89.66</b>	<b>76.15</b>
Word	64K	1.54 / 8.86	64.06	65.83	69.00	60.41	70.10	70.58	58.25
									55.30

Table 6: Performance of various models on several Korean natural language understanding tasks. The evaluation metrics are as follows: KorQuAD: Exact Match/F1, KorNLI: accuracy (%), KorSTS:  $100 \times$  Spearman correlation, NSMC: accuracy (%), PAWS-X: accuracy (%). The best scores in each column (global) and group (local) are bold-faced and underlined, respectively.

pairs—5,749 for training, 1,500 for development, and 1,379 for test. The task assesses the gradations of semantic similarity between two sentences with a scale from 0 to 5.

#### 5.2.4 Sentiment Analysis: NSMC Dataset

NSMC<sup>14</sup> is a movie review dataset scraped from Naver Movies™. It consists of 200K samples of which 150K are the training set and the rest 50K are the test set. Each sample is labeled with 0 (negative) or 1 (positive). We hold out 10 percent of the training data for development.

#### 5.2.5 Paraphrase Identification: PAWS-X Dataset

The PAWS-X dataset (Yang et al., 2019) is a challenging paraphrase identification dataset in six languages including Korean. The Korean portion amounts to 53,338 sentence pairs (49,410 for training, 1,965 for development, and 1,972 for test). Like the NSMC dataset, each sentence pair is annotated with either 0 (negative) or 1 (positive).

For each tokenization strategy, we pre-train a BERT-Base model on a large corpus and fine-tune it on the training sets of the five NLU tasks independently.

**Pre-training.** Because the Korean Wiki corpus is not enough in volume, 640 MB, for the pre-

training purpose, we additionally download the recent dump of Namuwiki<sup>15</sup>, a Korean Wiki, and extract plain texts using Namu Wiki Extractor<sup>16</sup>. On the resulting Namuwiki corpus (5.5 GB) along with the Wiki corpus (640 MB), pre-training is performed with a Cloud TPU v3-8 for 1M steps using the official BERT training code<sup>17</sup>, which is based on TensorFlow. We set the training hyper-parameters of all models as follows: batch size = 1024, max sequence length = 128, optimizer = AdamW (Loshchilov and Hutter, 2019), learning rate = 5e-5, warm-up steps = 10K.

**Fine-tuning.** After converting each of the pre-trained models in TensorFlow into PyTorch, we fine-tune it using HuggingFace Transformers<sup>18</sup> (Wolf et al., 2019). The hyper-parameters for each task are shown in Table 5.

#### 5.2.6 Results

In Table 6 we report the evaluation results of the various models on the dev and test sets. Since KorQuAD lacks the test set, we report the results on the dev set only.

<sup>15</sup><http://dump.thewiki.kr>

<sup>16</sup><https://github.com/jonghwanyeon/namu-wiki-extractor>

<sup>17</sup><https://github.com/google-research/bert>

<sup>18</sup><https://github.com/huggingface/transformers>

<sup>14</sup><https://github.com/e9t/nsmc>

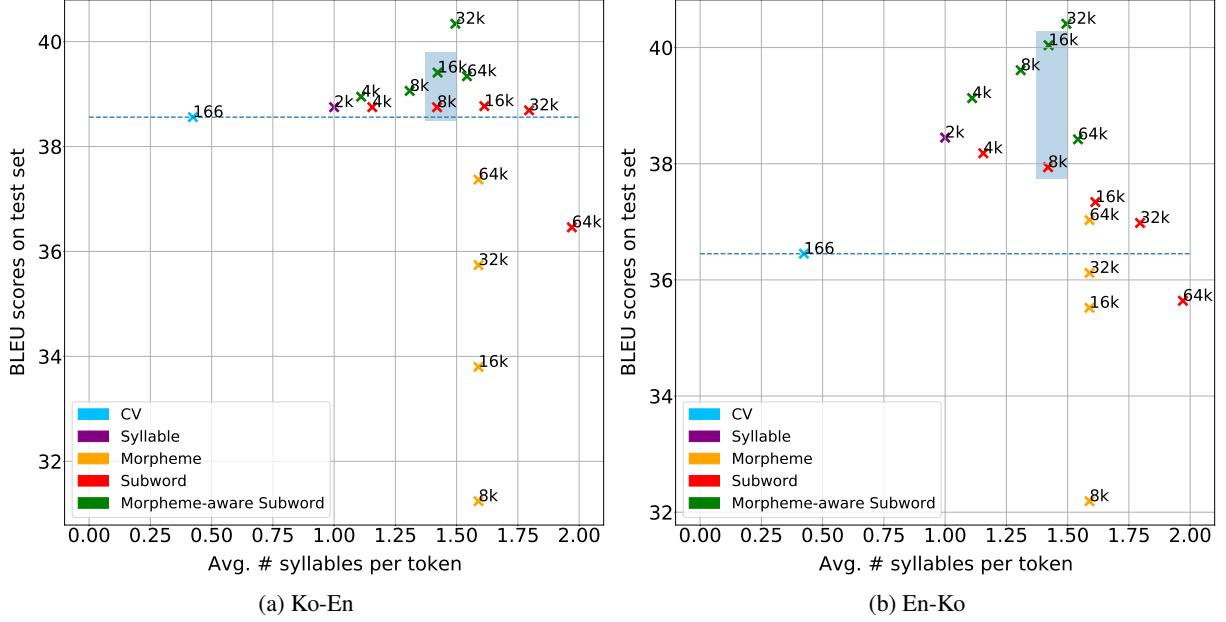


Figure 1: Translation performance over the average number of syllables per token

As for KorQuAD, Subword 64K models achieve the highest Exact Match (EM) and F1 scores. The scores in the Subword and Morpheme models increase monotonically as the vocabulary size grows. On the other hand, the 32K models outperform the others in the Morpheme-aware Subword models; no clear correlation is found between performance and vocabulary sizes in them.

For all the other four tasks, Morpheme-aware Subword 64K models show the best scores. One noteworthy phenomenon is that the scores tend to increase as the vocabulary size grows across the tokenization groups. This is discordant with the machine translation results in Section 5.1.4, where a larger vocabulary size does not guarantee better performance for the Subword and Morpheme-aware Subword models.

## 6 Discussion

We further examine which factors with respect to tokenization affect the Ko-En and En-Ko translation performance.

### 6.1 Token Length

Because tokenization involves splitting a text into shorter segments, we find it important to figure out how much information each segment bears. To this end, based on the assumption that the longer a text is, the more information it is likely to have, we plot the BLEU scores by the average number of syllables per Korean token in the translation test

sets in Figure 1.

The BLEU scores of the subword models—Syllable, Morpheme, Subword, and Morpheme-aware Subword—are mostly higher than those of the CV models, which are plotted as dotted lines. In particular, the Syllable, Subword, and Morpheme-aware Subword models between 1.00 and 1.50 show the best scores both in Ko-En and in En-Ko. When a token has more than 1.5 syllables on average, the scores begin to decrease, and the Word models which has more than 2.5 syllables in a token performs the worst (7.07 for Ko-En and 18.42 for En-Ko). Note that they are not in the figures due to space constraints.

### 6.2 Linguistic Awareness

Obviously token length is not the only key factor in tokenization strategies. Let us compare the Morpheme-aware Subword 16K models (green markers) and Subword 8K models (red markers) in the shaded regions in Figure 1. Although they have the same average token length around 1.4, the Morpheme-aware Subword models outperform the Subword models. We believe this is evidence to support that linguistic awareness is another important factor in Korean tokenization strategies for machine translation.

### 6.3 Under-trained Tokens

In section 5.1.4, we pointed out high OOV rates are highly likely to degrade the performance of

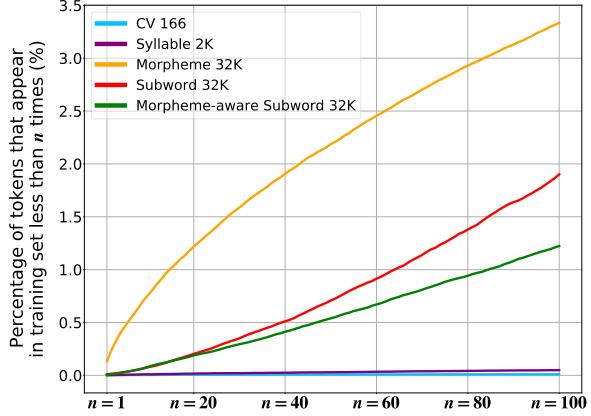


Figure 2: Percentage of under-trained tokens in various tokenization strategies

Morpheme models. It is also worth noting that in Figure 1 as most of the orange markers denoting Morpheme models are below the dotted lines.

OOVs are the tokens that appear only in the test set. They are an extreme case of under-trained tokens—test set’s tokens that appear in the training set for the limited number of times. Figure 2 shows how much under-trained tokens account for in each model, ranging from  $n = 1$  to  $n = 100$ , where  $n$  is the frequency of the under-trained tokens in the training set. Clearly, the curve of the Morpheme 32K model is far above that of the others, indicating that it suffers from the problem of under-trained tokens the most.

## 7 Conclusion

We explored various Korean tokenization strategies on machine translation and five NLU tasks. In machine translation Morpheme-aware Subword models with a vocabulary size worked best for both Korean to English and English to Korean settings. By contrast, there was no single best tokenization strategy for the NLU tasks. Instead, Subword 64K models showed the best performance on KorQuAD, whereas Morpheme-aware Subword 64K models turned out to be optimal for the other KorNLI, KoSTS, NSMC, and PAWS-X tasks.

## Acknowledgments

We are grateful to the anonymous reviewers for their valuable comments. For pre-training models, we used Cloud TPUs provided by TensorFlow Research Cloud program.

## References

- Tamali Banerjee and Pushpak Bhattacharyya. 2018. Meaningless yet meaningful: Morphology grounded subword-level NMT. In *Proceedings of the Second Workshop on Subword/Character LEvel Models*, pages 55–60, New Orleans. Association for Computational Linguistics.
- Kaj Bostrom and Greg Durrett. 2020. Byte pair encoding is suboptimal for language model pretraining. *arXiv preprint arXiv:2004.03720*.
- Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.
- Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.
- Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- Philip Gage. 1994. A new algorithm for data compression. *C Users J.*, 12(2):23–38.
- Jiyeon Ham, Yo Joong Choe, Kyubyong Park, Ilji Choi, and Hyungjoon Soh. 2020. KorNLI and KorSTS: New benchmark datasets for korean natural language understanding. *arXiv preprint arXiv:2004.03289*.
- Beom-mo Kang and Hung-gyu Kim. 2001. 21st century sejong project-compiling korean corpora. In *Proceedings of the 19th International Conference on Computer Processing of Oriental Languages (ICCPOL 2001)*, pages 180–183.
- Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *3rd International Conference on Learning Representations, ICLR 2015*.

- Taku Kudo. 2006. Mecab: Yet another part-of-speech and morphological analyzer. <https://sourceforge.net/projects/mecab/>.
- Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.
- Seungyoung Lim, Myungji Kim, and Jooyoul Lee. 2019. KorQuAD 1.0: Korean qa dataset for machine reading comprehension. *arXiv preprint arXiv:1909.07005*.
- Pierre Lison and Jörg Tiedemann. 2016. OpenSubtitles2016: Extracting large parallel corpora from movie and TV subtitles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 923–929, Portorož, Slovenia. European Language Resources Association (ELRA).
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.
- Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. *7th International Conference on Learning Representations, ICLR 2019*.
- Sangwhan Moon and Naoaki Okazaki. 2020. Jamo pair encoding: Subcharacter representation-based extreme Korean vocabulary compression for efficient subword tokenization. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 3490–3497, Marseille, France. European Language Resources Association.
- Toshiaki Nakazawa, Chenchen Ding, Hideya Mino, Isao Goto, Graham Neubig, and Sadao Kurohashi. 2016. Overview of the 3rd workshop on Asian translation. In *Proceedings of the 3rd Workshop on Asian Translation (WAT2016)*, pages 1–46, Osaka, Japan. The COLING 2016 Organizing Committee.
- Toshiaki Nakazawa, Nobushige Doi, Shohei Higashiyama, Chenchen Ding, Raj Dabre, Hideya Mino, Isao Goto, Win Pa Pa, Anoop Kunchukuttan, Shantipriya Parida, Ondřej Bojar, and Sadao Kurohashi. 2019. Overview of the 6th workshop on Asian translation. In *Proceedings of the 6th Workshop on Asian Translation*, pages 1–35, Hong Kong, China. Association for Computational Linguistics.
- Toshiaki Nakazawa, Shohei Higashiyama, Chenchen Ding, Hideya Mino, Isao Goto, Hideto Kazawa, Yusuke Oda, Graham Neubig, and Sadao Kurohashi. 2017. Overview of the 4th workshop on Asian translation. In *Proceedings of the 4th Workshop on Asian Translation (WAT2017)*, pages 1–54, Taipei, Taiwan. Asian Federation of Natural Language Processing.
- Toshiaki Nakazawa, Hideya Mino, Isao Goto, Graham Neubig, Sadao Kurohashi, and Eiichiro Sumita. 2015. Overview of the 2nd workshop on Asian translation. In *Proceedings of the 2nd Workshop on Asian Translation (WAT2015)*, pages 1–28, Kyoto, Japan. Workshop on Asian Translation.
- Toshiaki Nakazawa, Katsuhito Sudoh, Shohei Higashiyama, Chenchen Ding, Raj Dabre, Hideya Mino, Isao Goto, Win Pa Pa, Anoop Kunchukuttan, and Sadao Kurohashi. 2018. Overview of the 5th workshop on Asian translation. In *Proceedings of the 32nd Pacific Asia Conference on Language, Information and Computation: 5th Workshop on Asian Translation: 5th Workshop on Asian Translation*, Hong Kong. Association for Computational Linguistics.
- Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.
- Chanjun Park, Gyeongmin kim, and Heuiseok Lim. 2019a. Parallel corpus filtering and korean-optimized subword tokenization for machine translation. In *Proceedings of the 31st Annual Conference on Human & Cognitive Language Technology*.
- Cheoneum Park, Young-Jun Jung, Kihoon Kim, Geonyeong Kim, Jae-Won Jeon, Seongmin Lee, Junseok Kim, and Changki Lee. 2019b. KNU-HYUNDAI’s NMT system for scientific paper and patent tasks on WAT 2019. In *Proceedings of the 6th Workshop on Asian Translation*, pages 81–89, Hong Kong, China. Association for Computational Linguistics.
- Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.
- Mārcis Pinnis, Rihards Krišlauks, Daiga Deksne, and Toms Miks. 2017. Neural machine translation for morphologically rich languages with improved subword units and synthetic data. In *International Conference on Text, Speech, and Dialogue*, pages 237–245. Springer.
- Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. [Improving neural machine translation models with monolingual data](#). In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. [Neural machine translation of rare words with subword units](#). In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Ahmed Tawfik, Mahatab Emam, Khaled Essam, Robert Nabil, and Hany Hassan. 2019. [Morphology-aware word-segmentation in dialectal Arabic adaptation of neural machine translation](#). In *Proceedings of the Fourth Arabic Natural Language Processing Workshop*, pages 11–17, Florence, Italy. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. [Attention is all you need](#). In *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. [A broad-coverage challenge corpus for sentence understanding through inference](#). In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierrick Cistac, Tim Rault, R’emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. [Huggingface’s transformers: State-of-the-art natural language processing](#). *ArXiv*, abs/1910.03771.

Yinfei Yang, Yuan Zhang, Chris Tar, and Jason Baldridge. 2019. [PAWS-x: A cross-lingual adversarial dataset for paraphrase identification](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3687–3692, Hong Kong, China. Association for Computational Linguistics.

# BERT-Based Neural Collaborative Filtering and Fixed-Length Contiguous Tokens Explanation

Reinald Adrian Pugoy<sup>1,2</sup> and Hung-Yu Kao<sup>1</sup>

<sup>1</sup>National Cheng Kung University, Tainan City, Taiwan

<sup>2</sup>University of the Philippines Open University, Los Baños, Philippines

rdpugoy@up.edu.ph, hykao@mail.ncku.edu.tw

## Abstract

We propose a novel, accurate, and explainable recommender model (BENEFICT) that addresses two drawbacks that most review-based recommender systems face. First is their utilization of traditional word embeddings that could influence prediction performance due to their inability to model the word semantics' dynamic characteristic. Second is their black-box nature that makes the explanations behind every prediction obscure. Our model uniquely integrates three key elements: BERT, multilayer perceptron, and maximum subarray problem to derive contextualized review features, model user-item interactions, and generate explanations, respectively. Our experiments show that BENEFICT consistently outperforms other state-of-the-art models by an average improvement gain of nearly 7%. Based on the human judges' assessment, the BENEFICT-produced explanations can capture the essence of the customer's preference and help future customers make purchasing decisions. To the best of our knowledge, our model is one of the first recommender models to utilize BERT for neural collaborative filtering.

## 1 Introduction

In recommender systems research, collaborative filtering (CF) is the dominant state-of-the-art recommendation model, which primarily focuses on learning accurate representations of users (user preferences) and items (item characteristics) (Chen et al., 2018; Tay et al., 2018). The earliest recommender models learned these representations based on user-given numeric ratings that each item received (Mnih and Salakhutdinov, 2008; Koren et al., 2009). However, ratings, which are values on a single discrete scale, oversimplify user preferences and item characteristics (Musto et al., 2017). The large amount of users and items in a typical online platform consequently results in a highly

sparse rating matrix, making it hard to learn accurate representations (Zheng et al., 2017).

To alleviate these issues, review texts have instead been utilized to model such representations for subsequent recommendation and rating prediction, and this approach has attracted growing attention in research (Catherine and Cohen, 2017; Zheng et al., 2017). The main advantage of reviews as the source of features is that they can cover user opinions' multi-faceted substance. Because users can explain their reasons underlying their given ratings, reviews contain a large amount of latent information that is both rich and valuable, and that cannot be otherwise obtained from ratings alone (Chen et al., 2018; Wang et al., 2019). Recently, models that incorporate user reviews have yielded state-of-the-art performances (Zheng et al., 2017; Chen et al., 2018). These approaches learn user and item representations by using traditional word embeddings (e.g., word2vec, GloVe) to map each word in the review into its corresponding vector. The review is transformed into an embedded matrix before being fed to a convolutional neural network (CNN) (Chen et al., 2018). CNNs have been shown to effectively model reviews and have illustrated outstanding results in numerous natural language processing tasks (Wang et al., 2018a).

Nevertheless, there are drawbacks that most review-based recommender models experience. First is the utilization of traditional or mainstream word embeddings to learn review features. Their static nature is a hindrance, as each word sense is associated with the same embedding regardless of the context. In other words, such embeddings cannot identify the dynamic nature of each word's semantics. For review-based recommenders, this could be an issue in modeling users and items, which could, in turn, affect recommendation performance (Pilehvar and Camacho-Collados, 2019). Also, once a CNN is fed with the matrix of word embeddings, the word frequency information of contextual fea-

tures, said to be crucial for modeling reviews, will be lost (Wang et al., 2018a).

Another drawback is the inherent black-box nature of deep learning-based models that makes the explanations behind every prediction obscure (Ribeiro et al., 2016; Wang et al., 2018b). The complex architecture of hidden layers has opaqued the models’ internal decision-making processes (Peake and Wang, 2018). Providing explanations could help persuade users to make decisions and develop trust in a recommender system (Zhang et al., 2014; Ribeiro et al., 2016; Costa et al., 2018; Peake and Wang, 2018). However, this leads us to a dilemma, i.e., a trade-off between accuracy and explainability. Usually, the most accurate models are inherently complicated, non-transparent, and unexplainable (Zhang and Chen, 2018). The same is also true for explainable and straightforward methods that sacrifice accuracy. Formulating models that are both explainable and accurate is a challenging yet critical research agenda for the machine learning community to ensure that we derive benefits from machine learning fairly and responsibly (Peake and Wang, 2018).

In this paper, we propose a unique model: **BERT-Based Neural Collaborative Filtering and Fixed-Length Contiguous Tokens Explanation (BENEFICT)**. Our model learns user and item representations simultaneously using two parallel networks. To address the first drawback, we incorporate BERT as a key component in each parallel network. BERT affords us to extract more meaningful, contextualized features adaptable to arbitrary contexts; such features cannot be derived from mainstream word embeddings (Pilehvar and Camacho-Collados, 2019; Zaki et al., 2019). BERT can also retain the word frequency information that makes CNN an unnecessary component of our model. Once user and item representations are learned, they are concatenated together in a shared hidden space before being finally fed to an optimal stack of multilayer perceptron (MLP) layers that serve as BENEFICT’s interaction function.

To address the second drawback, we introduce a novel component in our model that integrates BERT’s self-attention and an implementation of the fixed-length maximum subarray problem (MSP), which is considered to be a classic computer science problem. BERT applies self-attention in each encoder layer that consequently produces self-attention weights for each token. These are passed

to the successive encoder layers through feedforward networks. We argue that these self-attention weights can be the basis for explaining rating predictions. Based on this premise, MSP then selects a segment or subarray of consecutive tokens that has the maximum possible sum of self-attention weights.

## 1.1 Contributions

Our work aims to fill the research gap by implementing a solution that is both accurate and explainable. We propose a novel model that uniquely integrates three vital elements, i.e., BERT, MLP, and MSP, to derive review features, model user-item interactions, and produce possible explanations. To the best of our knowledge, BENEFICT is one of the first review-based recommender models to utilize BERT for neural CF. Also, to the extent of our knowledge, BENEFICT is one of the first models to repurpose a portion of the Neural Collaborative Filtering (NCF) framework (He et al., 2017) as the user-item interaction function for review-based, explicit CF. Moreover, our experiments have demonstrated that our model achieves better rating prediction results than the other state-of-the-art recommender models.

## 2 Related Work and Concepts

Designing a CF model involves two crucial steps: learning user and item representations and modeling user-item interactions based on those representations (He et al., 2018). Before the advancements provided by neural networks, matrix factorization (MF) was the dominant model representing users and items as vectors of latent factors (called embeddings) and models user-item interactions using the inner product operation. The said operation leads to poor performance because it is sub-optimal for learning rich yet complicated patterns from real-world data (He et al., 2018). To address this scenario, neural networks (NN) have been integrated into recommender architectures. One of the initial works that have laid the foundation in employing NN for CF is NCF (He et al., 2017). Their framework, originally implemented for rating-based, implicit CF, learns non-linear interactions between users and items by employing MLP layers as their interaction function, granting it a high degree of non-linearity and flexibility to learn meaningful interactions. Two common designs have emerged when it comes to leveraging MLP layers: placing

an MLP above either the concatenated user-item embeddings (He et al., 2017; Bai et al., 2017) or the element-wise product of user and item embeddings (Zhang et al., 2017; Wang et al., 2017).

As far as rating prediction is concerned, two notable recommender models have yielded significant state-of-the-art prediction performances. DeepCoNN is the first deep model that represents users and items from reviews jointly (Zheng et al., 2017). It consists of two parallel, CNN-powered networks. One network learns user behavior by examining all reviews that he has written, and the other network models item properties by exploring all reviews that it has received. A shared layer connects these two networks, and factorization machines capture user-item interactions. The second model is NARRE, which shares certain similarities with DeepCoNN. NARRE is also composed of two parallel networks for user and item modeling with respective CNNs to process reviews (Chen et al., 2018). Rather than concatenating reviews to one long sequence the same way that DeepCoNN does, their model introduces an attention mechanism that learns review-level usefulness in the form of attention weights. These weights are integrated into user and item representations to enhance the embedding quality and the subsequent prediction accuracy. Both DeepCoNN and NARRE employ traditional word embeddings.

Other relevant studies have claimed to provide explanations for recommendations such as EFM (Zhang et al., 2014), sCVR (Ren et al., 2017), and TriRank (He et al., 2015). These models initially extract aspects and opinions by performing phrase-level sentiment analysis on reviews. Afterward, they generate feature-level explanations according to product features that correspond to user interests (Chen et al., 2018). However, these models have some limitations; manual preprocessing is required for sentiment analysis and feature extraction, and the explanations are simple extraction of words or phrases from the review text (Zhang et al., 2014; Ren et al., 2017). This also has the unintended effect of distorting the reviews’ original meaning (Ribeiro et al., 2016; Chen et al., 2018). Another limitation is that textual similarity is solely based on lexical similarity; this implies that semantic meaning is ignored (Zheng et al., 2017; Chen et al., 2018).

### 3 Methodology

BENEFICT, as illustrated in Figure 1, has two parallel networks to model user and item embeddings that both utilize BERT. Hereafter, we will only illustrate the user modeling process because the same is also observed for item modeling, with their inputs as the only difference.

#### 3.1 Input Layer and BERT Encoding

Given an input set of user-written reviews  $V_u = \{V_{u1}, V_{u2}, \dots, V_{uj}\}$  where  $j$  is the total number of reviews from user  $u$ ,  $V_u$  is fed to a pre-trained BERT<sub>BASE</sub> model to encode the reviews and obtain their respective contextualized representations. BERT<sub>BASE</sub> consists of 12 encoder layers and 12 self-attention heads (Devlin et al., 2018). It also has a hidden size of 768, which we will directly utilize later as the fixed embedding dimension. Furthermore, BERT requires every review to follow a particular format. For this purpose, the model applies WordPiece tokenization to the review’s input sequence (Wu et al., 2016). The format is comprised of token embeddings, segment embeddings, position embeddings, and padding masks. Because rating prediction is not a sentence pairing task, BERT takes each review as a single segment of contiguous text. Typically, BERT supports a maximum sequence length of 512 tokens. In this study, we use a shorter length of 256 tokens to save substantial memory. As such, each input sequence is truncated or padded accordingly.

The newly-formatted input sequence then passes through a stack of Transformer encoders to obtain the contextualized representations of reviews:  $h_{[\text{CLS}],u} = \{h_{[\text{CLS}],u1}, h_{[\text{CLS}],u2}, \dots, h_{[\text{CLS}],uj}\}$ , where  $h_{[\text{CLS}],u} \in \mathbb{R}^{j \times 768}$ . We utilize the hidden state of the special [CLS] token to serve as the review’s aggregate sequence representation or pooled contextualized embedding (Devlin et al., 2018). In theory, any encoder layer may be selected to provide the hidden state of [CLS] as the review’s representation. We select the twelfth layer for our approach; prior studies have illustrated that its predictive capability is the best among the other layers (Sun et al., 2019).

#### 3.2 Embedding Generation, Multilayer Perceptron, and Prediction

The user embedding (user feature vector)  $P_u \in \mathbb{R}^{1 \times 768}$  is obtained by calculating the average of the [CLS] representations of the reviews written by

user  $u$ , given by the formula below. Similarly, the item embedding (item feature vector)  $Q_i \in \mathbb{R}^{1 \times 768}$  can be generated from the item modeling network.

$$P_u = \frac{1}{j} \sum_{t=1}^j h_{[\text{CLS}],ut} \quad (1)$$

Furthermore, the purpose of incorporating an MLP is to learn the interactions between user and item representations and to model the CF effect, which will not be properly covered by solely using vector concatenation or element-wise product (He et al., 2017). Adding a certain number of hidden layers on top of the concatenated user-item embedding provides further flexibility and non-linearity. Formally, the MLP component of BENEFICT is defined as follows:

$$\begin{aligned} h_0 &= [P_u, Q_i]^T \\ h_1 &= \text{ReLU}(W_1 h_0 + b_1) \\ h_L &= \text{ReLU}(W_L h_{L-1} + b_L) \\ \hat{R}_{ui} &= W_{L+1} h_L + b_{L+1} \end{aligned} \quad (2)$$

where  $h_0 \in \mathbb{R}^{1 \times 1536}$  is the concatenated user-item embedding in the shared hidden space;  $h_L$  represents the  $L$ -th MLP layer;  $W_L$  and  $b_L$  pertain to the weight matrix and bias vector of the  $L$ -th layer, respectively; and  $\hat{R}_{ui}$  denotes the predicted rating that user  $u$  gives to item  $i$ . For the activation function of the MLP layers, we choose the rectified linear unit (ReLU), which generally yields better performance than other activation functions such as tanh and sigmoid (Glorot et al., 2011; He et al., 2016, 2017).

Concerning the structure, our model’s MLP component follows a tower pattern where the bottom layer is the widest, and every subsequent top layer has a smaller number of neurons. The rationale behind this is that the MLP can learn more abstractive data features by utilizing fewer hidden units for the top layers (He et al., 2016). In our implementation for a three-layered MLP, the number of neurons from the bottom layer to the top layer follows this pattern: 1536 (concatenated embedding)  $\rightarrow$  768 (MLP layer 1)  $\rightarrow$  384 (MLP layer 2)  $\rightarrow$  192 (MLP layer 3)  $\rightarrow$  1 (prediction layer)

### 3.3 Learning

In training the model, the loss function is the mean squared error (MSE) given by this formula:

$$MSE = \frac{1}{|Tr|} \sum_{u,i \in Tr} (R_{ui} - \hat{R}_{ui})^2 \quad (3)$$

where  $Tr$  refers to the training samples or instances, and  $R_{ui}$  is the ground-truth rating given by user  $u$  to item  $i$ . Moreover, we employ the Adaptive Moment Estimation with weight decay or AdamW (Loshchilov and Hutter, 2018) to optimize the loss function. Based on the original Adam optimizer, AdamW also leverages the power of adaptive learning rates during training. This makes the selection of a proper learning rate less cumbersome that consequently leads to faster convergence (Chen et al., 2018). Unlike Adam, AdamW implements a weight decay fix, a regularization technique that prevents weights from growing too huge and is proven to yield better training loss and generalization error (Loshchilov and Hutter, 2018).

### 3.4 Explanation Generation

The stack of BERT’s Transformer encoders also provides sets of self-attention weights that a token gives to every token found in the review text. We are particularly interested in the attention that  $[\text{CLS}]$  gives to each review token using the twelfth layer’s multiple attention heads. Given an input sequence of tokens  $F_{uj}$  produced by WordPiece tokenization from review  $V_{uj}$ , a set of attention weights is represented as:

$$\alpha_{[\text{CLS}],uj} = \{\alpha_1^k(F_{uj}), \alpha_2^k(F_{uj}), \dots, \alpha_g^k(F_{uj})\} \quad (4)$$

where  $k$  is the specific attention head in a particular encoder layer, and  $\alpha_g^k$  is the attention that  $[\text{CLS}]$  gives to the  $g$ -th WordPiece token over the input sequence  $F_{uj}$ . There are 12 attention heads in an encoder layer which translate to 12 different attention weights that each token receives from the  $[\text{CLS}]$  token. For a given token  $g$ , the following formula is applied to compress the weights into a single value:

$$ComAtt_g = \sum_{k=1}^{12} \alpha_g^k(F_{uj}) \quad (5)$$

We then reformulate the task of generating explanations as a fixed-length MSP. In its vanilla sense, MSP selects a segment of consecutive array elements (i.e., a contiguous subarray of tokens) that has the maximum possible sum over all other segments (Bae, 2007). In this paper, we introduce constraint  $N$  to the MSP;  $N$  is a fixed value that pertains to the length of the explanation. Formally, the set of compressed attention weights per review is given by the following array:

$$A_{uj} = [ComAtt_1, ComAtt_2, \dots, ComAtt_g] \quad (6)$$

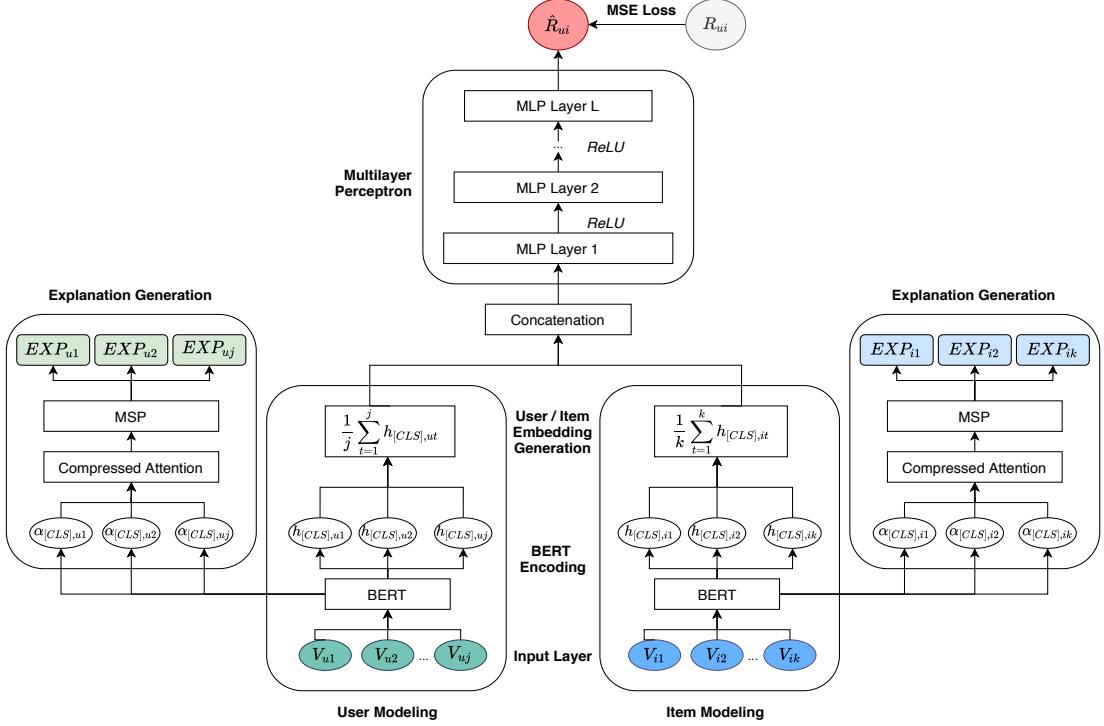


Figure 1: The proposed BENEFICT architecture.

Dataset	#Reviews	#Users	#Items
Toys and Games	167,597	19,412	11,924
Digital Music	64,706	5,541	3,568
Yelp-Dense	159,114	8,919	7,122
Yelp-Sparse	229,907	45,981	11,537

Table 1: Statistics summary of the datasets.

The goal is to find token indices  $x$  and  $y$  that maximize:

$$\sum_{t=x}^y A_{uj}[t] \quad (7)$$

This is subject to the requirements that  $1 \leq x < y \leq 256$  and  $(y - x) + 1 = N$ . Finally, the generated explanation for review  $V_{uj}$  is represented as:

$$EXP_{uj} = \text{Concat}(F_{uj,x}, F_{uj,x+1}, \dots, F_{uj,y}) \quad (8)$$

## 4 Experiments

In this section, we perform relevant experiments intending to answer the following research questions:

**RQ1:** Does BENEFICT outperform other state-of-the-art recommender models?

**RQ2:** What is the optimal configuration for learning user-item interactions?

**RQ3:** Can our model produce explanations acceptable to humans?

### 4.1 Datasets and Experimental Settings

Table 1 summarizes the four public datasets from different domains used in our study. Two of these datasets are Amazon 5-core<sup>1</sup>: **Toys and Games**, which consists of nearly 168 thousand reviews, and **Digital Music**, which contains about 65 thousand reviews (McAuley et al., 2015). These datasets are said to be 5-core wherein users and items have five reviews each. We also utilize **Yelp**<sup>2</sup>, a large-scale dataset for restaurant feedback and ratings. We both employ its original, sparse version and its 5-core, dense version with about 160 thousand and 230 thousand reviews, respectively. The ratings in all datasets are in the range of [1, 5]. We randomly split each dataset of user-item pairs into training (80%), validation (10%), and test (10%) sets. In our experiments, we perform an exhaustive grid search over the following hyperparameters: number of epochs [1, 20] and number of MLP layers [0, 3]. The learning rate and weight decay are both set to

<sup>1</sup><http://jmcauley.ucsd.edu/data/amazon/>

<sup>2</sup><https://github.com/danielfrg/kaggle-yelp-recruiting-competition>

Model	Toys and Games	Digital Music	Yelp-Dense	Yelp-Sparse	Average
DeepCoNN	0.8971	0.8972	1.0311	1.2006	1.0065
NARRE	0.8840	0.8997	1.0312	1.1770	0.9979
<b>BENEFICT</b>	<b>0.8348</b>	<b>0.8750</b>	<b>0.9963</b>	<b>0.9764</b>	<b>0.9206</b>
$\Delta$ BENEFICT	5.57%	2.47%	3.38%	17.04%	7.11%

Table 2: RMSE comparison of the recommender models. The best RMSE values are highlighted in bold. The last row shows the improvement gained by BENEFICT against the better performing baseline.

0.001. Due to memory limitations, the batch size is fixed at 32. We select the model configuration (i.e., a grid point) with the best root mean square error (RMSE) on the validation set. We use the test set for evaluating the model’s final performance.

#### 4.2 Baselines and Evaluation Metric

To validate the effectiveness of BENEFICT, we select two other state-of-the-art models as baselines:

- **DeepCoNN** (Zheng et al., 2017): It is a deep collaborative neural network model based on two parallel CNNs to learn user and item feature vectors in a joint manner.
- **NARRE** (Chen et al., 2018): Similar to DeepCoNN, it is a neural attentional regression model that integrates two parallel CNNs and an attention mechanism to model latent features.

Afterward, we calculate the RMSE, a widely used metric for rating prediction, to evaluate the models’ respective performances.

$$RMSE = \sqrt{\frac{1}{|Ts|} \sum_{u,i \in Ts} (R_{ui} - \hat{R}_{ui})^2} \quad (9)$$

In the formula,  $Ts$  denotes the test samples or instances of user-item pairs.

#### 4.3 Prediction Results and Discussion

Table 2 reports the RMSE values of BENEFICT and the two baselines, with the last row (represented by  $\Delta$ BENEFICT) indicating the improvement gained by our model compared with the better baseline. The results show that BENEFICT consistently outperforms the baselines across all datasets; our model has an average RMSE score of 0.9206, as opposed to 1.0065 and 0.9979 for DeepCoNN and NARRE, respectively. On average, this has

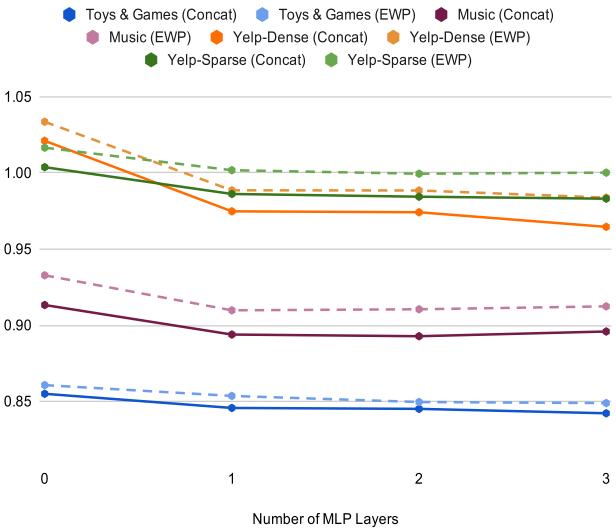


Figure 2: RMSE comparison of BENEFICT variants using different user-item interaction functions. The solid lines pertain to the concatenation-MLP interaction function. On the other hand, the broken lines refer to the interaction function based on the element-wise product (EWP) and MLP.

resulted in the improvement gained by BENEFICT of nearly 7%. These results validate our hypothesis that using BERT-derived embeddings and representations, considered to be more semantically meaningful than their traditional counterparts, can significantly improve rating prediction accuracy and that BERT can likewise offset the limitations of mainstream word embeddings and CNN.

Moreover, the rationale of employing two versions of Yelp is to compare the recommender models’ performances on both dense and sparse datasets. As illustrated in the fourth and fifth columns of Table 2, both the RMSE values of DeepCoNN and NARRE worsen when they attempt to perform predictions on the original, sparse Yelp. For DeepCoNN, from the dense version’s RMSE of 1.0311, it increases to 1.2006. The same is

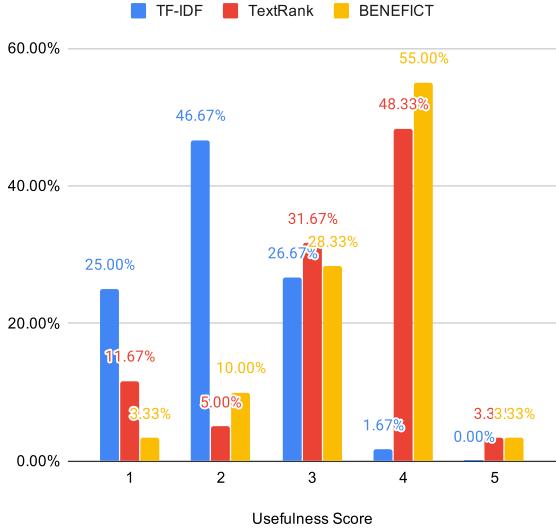


Figure 3: Distribution of the judges’ given usefulness scores based on US1.

also true for NARRE, whose RMSE increases to 1.1770 from 1.0312. Interestingly, BENEFICT produces an entirely different observation; its RMSE decreases to 0.9764 from 0.9963. Our model’s improvement is 17.04%, greater than  $\Delta$ BENEFICT for the three other datasets. We attribute these findings to the greater amount of information in Yelp-Sparse that can be successfully utilized by BENEFICT for modeling reviews. It should be noted that Yelp-Sparse has nearly 230 thousand reviews, while Yelp-Dense has almost 160 thousand. In conclusion, these results provide evidence that our model is best equipped and capable of performing predictions regardless of a dataset’s inherent sparsity or density.

### 4.3.1 Optimal Interaction Function

BENEFICT employs an MLP above the concatenated user-item embeddings in the shared hidden space. We compare it against another variant of our model, which utilizes an MLP on top of the element-wise product of user and item representations. We examine their performances using a different number of hidden layers [0, 3]. It should be noted that an MLP with zero layers pertains to the shared hidden space’s direct projection to the prediction layer.

Figure 2 demonstrates that BENEFICT’s utilization of concatenation exceeds the element-wise product by a significant margin across all MLP layers and datasets. This result verifies the positive effect of feeding the concatenated features

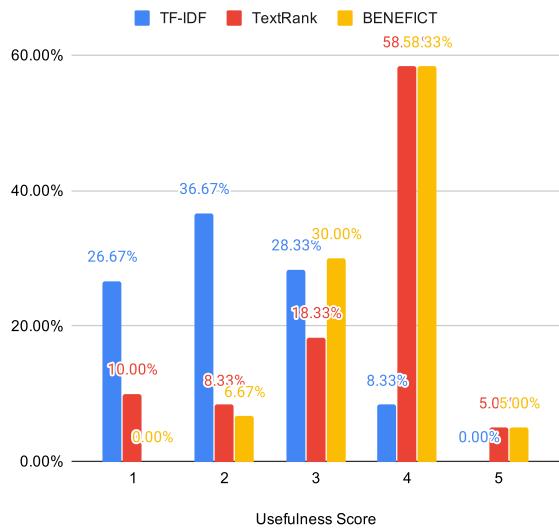


Figure 4: Distribution of the judges’ given usefulness scores based on US2.

to the MLP to learn user-item interactions. Furthermore, consistent with the findings of He et al. (2017), stacking more layers is indeed beneficial and effective for neural explicit collaborative filtering as well. There appears to be a trend: increasing the hidden layers implies decreasing (and better) RMSE values. Simply projecting the shared hidden space to the prediction layer is insufficient and weak, as evidenced by its relatively high RMSE scores. On the contrary, using three MLP layers has generally resulted in the lowest RMSE scores. The only exception is with the Digital Music dataset wherein utilizing two layers produces the best RMSE value. Furthermore, even though the element-wise product is more inferior than concatenation, the former also benefits from increasing the MLP layers. In summary, all these findings validate the necessity of incorporating the MLP as an integral part of the whole BENEFICT model.

## 5 Explainability Study

### 5.1 Human Assessment of Explanations

To validate the helpfulness of BENEFICT-produced explanations in real life, we also generate possible explanations using **TF-IDF** and **TextRank**. Applying TF-IDF determines which words are more favorable or relevant in a corpus of documents (Rajaraman and Ullman, 2011). To make the assessment fair, we only select words with the top  $N$  TF-IDF scores, where the value of  $N$  is the same as the constraint introduced in BENEFICT’s

Explanation	US Scores
<b>TF-IDF:</b> Some of the tracks were really quite ... dare I say it, catchy. And there was even a Top 30-friendly single on the album ('Only Time will tell'). But wasn't this Carl Palmer – he of the 70s triple album and serious devotee of classical percussionist James Blades? And wasn't this also Steve Hose – he of another 70s triple album and several serious solo albums. And hadn't John Wetton starred on the seriously serious 'Red' in 74? How could the three come together yet produce this Adult-Oriented stadium Rock? Let's not forget Palmer's beginnings in the Crazy World of Arthur Brown and Atomic Rooster. Or Wetton's bizarre phase with Uriah Heep. And Geoff Downes was nominally half of 'Buggles', whose minimal output was unashamed pop. The style of this, Asia's debut album wasn't a million miles from UK's eponymous LP of 1978, although it was distinctly more mainstream. I like this album, the best of all the Asia output that I've heard. I would have preferred the music to be a little more ambitious; there's a sense in which it's all been concocted to maximise the commercial return, which you couldn't say of UK. But it's a good, undemanding listen.	<b>US1:</b> 1.5 <b>US2:</b> 1.5
<b>TextRank:</b> Some of the tracks were really quite ... dare I say it, catchy. And there was even a Top 30-friendly single on the album ('Only Time will tell'). But wasn't this Carl Palmer – he of the 70s triple album and serious devotee of classical percussionist James Blades? And wasn't this also Steve Hose....	<b>US1:</b> 2 <b>US2:</b> 2
<b>BENEFICT:</b> ....The style of this, Asia's debut album wasn't a million miles from UK's eponymous LP of 1978, although it was distinctly more mainstream. I like this album, the best of all the Asia output that I've heard. I would have preferred the music to be a little more ambitious; there's a sense in which it's all been concocted to maximise the commercial return, which you couldn't say of UK.....	<b>US1:</b> 4 <b>US2:</b> 4

Table 3: Sample explanations (highlighted in yellow) generated by TF-IDF, TextRank, and BENEFICT from a specific user review. The second column includes the average judge-given US1 and US2 scores.

explanation generation module. On the other hand, TextRank is a fully unsupervised, graph-based extractive summarization algorithm (Mihalcea and Tarau, 2004). Its goal is to rank entire sentences that comprise a given review text. Also, to make the assessment consistent, we only take the top sentence with a length of less than or equal to  $N$  for each review.

We then ask two human judges to evaluate a total of 90 explanations, 30 explanations each for TF-IDF, TextRank, and BENEFICT, with  $N = 20$ . We instruct them to score each explanation based on the following usefulness statements (US) on a five-point Likert scale, ranging from 1 (strongly disagree) to 5 (strongly agree).

**US1:** The explanation captures the essence of the customer's preference (like or dislike) in the review.

**US2:** The explanation is helpful for you or any customer to decide to purchase that particular item in the future.

We further examine the human assessment results by determining the strength of agreement between the two judges. This is done by calculating

the Quadratic Weighted Kappa (QWK) statistic. It measures inter-rater agreement and is suitable for ordinal or ranked variables. The Kappa metric lies on a scale of -1 to 1, where 1 implies perfect agreement, 0 indicates random agreement, and negative values mean that the agreement is less than chance, such as disagreement. Specifically, a coefficient of 0.01-0.20 indicates slight agreement, 0.21-0.40 implies fair agreement, 0.41-0.60 refers to moderate agreement, 0.61-0.80 pertains to substantial agreement, and 0.81-0.99 denotes nearly perfect agreement (Borromeo and Toyama, 2015).

## 5.2 Explainability Results and Discussion

### 5.2.1 Overall Assessment

Figure 3 summarizes the judges' given scores on their assessment of explanations based on US1. They find that nearly 58% of BENEFICT-derived explanations capture the essence of the customer's preference (i.e., those with usefulness scores of either four or five). It is followed by TextRank, with almost 52% of its produced explanations, and TF-IDF, with only 1.67% of its generated explanations. With respect to the inter-rater agreement on US1

in Table 5, the judges express fair agreement on BENEFICT (having a Kappa value of 0.2019). On the other hand, they slightly agree with each other on both TF-IDF and TextRank, with QWK values of 0.1924 and 0.0625, respectively. As Table 4 indicates, our model has a mean usefulness score of 3.45, better than TextRank (3.26) and TF-IDF (2.05).

Figure 4 shows the judges’ assessment scores based on US2. Interestingly, the judges express that nearly 63% of the explanations generated by BENEFICT and TextRank are helpful for any future customers. Upon including the low-scoring explanations, BENEFICT is still better than TextRank; the former has a mean usefulness score of 3.61 against the latter’s 3.40. Furthermore, the judges moderately agree as far as our model’s generated explanation is concerned (with a Kappa value of 0.4705). At the same time, they express less than chance agreement for TextRank (obtaining a Kappa value of -0.0073). This statement means that the large majority of TextRank’s high assessment scores come from one judge alone. Lastly, the judges observe that only 8.33% of the explanations from TF-IDF are helpful, with a mean usefulness score of 2.18 and a QWK value of 0.1921, which implies their slight agreement.

These results indicate that BENEFICT’s explanation generation module can effectively provide useful explanations that capture the essence of the customer’s preference and help future customers make purchasing decisions.

### 5.2.2 Specific Example Comparison

Given an example, we highlight words that serve as the explanations in Table 3. The explanation produced by TF-IDF can capture a few important words, such as *unashamed* and *undemanding*. However, due to its bag-of-words property, it includes several other unnecessary words that may not contribute to the explanation. Therefore, the judges do not find it to be helpful. Next, the TextRank-generated explanation also does not appear to capture the essence of the user’s like or dislike. It does not seem useful for customers to decide whether to purchase that item in the future. Still, the judges give TextRank higher usefulness scores than TF-IDF, even though the latter captures more adjectives and important words. We attribute this to human’s natural bias toward less noisy sentences that express complete thoughts. Lastly, the BENEFICT-produced explanation con-

Method	US1 Mean	US2 Mean
TF-IDF	2.05	2.18
TextRank	3.26	3.40
BENEFICT	3.45	3.61

Table 4: Mean usefulness scores of explanations assessed by the judges, based on US1 and US2.

Method	US1 QWK	US2 QWK
TF-IDF	0.1924	0.1921
TextRank	0.0625	-0.0073
BENEFICT	0.2019	0.4705

Table 5: The strength of inter-judge agreement for both US1 and US2 given by the QWK values.

veys a near-complete thought; take note that it is not a sentence but a segment of contiguous tokens that maximize the sum of attention weights. This enables BENEFICT to capture important phrases such as *like this album* and *the best of all*. Hence, the judges agree that it captures the essence of the customer’s preference and helps customers make purchasing decisions in the future.

## 6 Conclusion and Future Work

We have successfully implemented a novel recommender model that uniquely integrates BERT, MLP, and MSP. BENEFICT’s predictive capability is validated by experiments performed on Amazon and Yelp datasets, consistently outperforming other state-of-the-art models. Moreover, its explanation generation capability is verified by human judges. We argue that our work offers an avenue to help bridge the research gap between accuracy and explainability. In the future, we will consider incorporating other neural components, such as attention mechanisms, in improving the user-item modeling process. We also intend to enhance the expressiveness and the overall quality of the generated explanations.

## Acknowledgment

First and foremost, we extend our gratitude to the hardworking anonymous reviewers for their valuable insights and suggestions. Likewise, we sincerely thank the judges in our explainability study, Dr. Ria Mae Borromeo and Ms. Verna Banasihan, for their time and participation.

## References

- Sung Eun Bae. 2007. Sequential and parallel algorithms for the generalized maximum subarray problem.
- Ting Bai, Ji-Rong Wen, Jun Zhang, and Wayne Xin Zhao. 2017. A neural collaborative filtering model with interaction-based neighborhood. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1979–1982.
- Ria Mae Borromeo and Motomichi Toyama. 2015. Automatic vs. crowdsourced sentiment analysis. In *Proceedings of the 19th International Database Engineering & Applications Symposium*, pages 90–95.
- Rose Catherine and William Cohen. 2017. Transnets: Learning to transform for recommendation. In *Proceedings of the eleventh ACM conference on recommender systems*, pages 288–296.
- Chong Chen, Min Zhang, Yiqun Liu, and Shaoping Ma. 2018. Neural attentional rating regression with review-level explanations. In *Proceedings of the 2018 World Wide Web Conference*, pages 1583–1592. International World Wide Web Conferences Steering Committee.
- Felipe Costa, Sixun Ouyang, Peter Dolog, and Aonghus Lawlor. 2018. Automatic generation of natural language explanations. In *Proceedings of the 23rd International Conference on Intelligent User Interfaces Companion*, page 57. ACM.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
- Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.
- Xiangnan He, Tao Chen, Min-Yen Kan, and Xiao Chen. 2015. Trirank: Review-aware explainable recommendation by modeling aspects. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 1661–1670. ACM.
- Xiangnan He, Xiaoyu Du, Xiang Wang, Feng Tian, Jin-hui Tang, and Tat-Seng Chua. 2018. Outer product-based neural collaborative filtering. *arXiv preprint arXiv:1808.03912*.
- Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*, pages 173–182.
- Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37.
- Ilya Loshchilov and Frank Hutter. 2018. Fixing weight decay regularization in adam.
- Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 43–52.
- Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411.
- Andriy Mnih and Russ R Salakhutdinov. 2008. Probabilistic matrix factorization. In *Advances in neural information processing systems*, pages 1257–1264.
- Cataldo Musto, Marco de Gemmis, Giovanni Semeraro, and Pasquale Lops. 2017. A multi-criteria recommender system exploiting aspect-based sentiment analysis of users’ reviews. In *Proceedings of the eleventh ACM conference on recommender systems*, pages 321–325. ACM.
- Georgina Peake and Jun Wang. 2018. Explanation mining: Post hoc interpretability of latent factor models for recommendation systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2060–2069. ACM.
- Mohammad Taher Pilehvar and Jose Camacho-Collados. 2019. Wic: the word-in-context dataset for evaluating context-sensitive meaning representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1267–1273.
- Anand Rajaraman and Jeffrey David Ullman. 2011. *Mining of massive datasets*. Cambridge University Press.
- Zhaochun Ren, Shangsong Liang, Piji Li, Shuaiqiang Wang, and Maarten de Rijke. 2017. Social collaborative viewpoint regression with explainable recommendations. In *Proceedings of the tenth ACM international conference on web search and data mining*, pages 485–494. ACM.

- Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144. ACM.
- Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification? *arXiv preprint arXiv:1905.05583*.
- Yi Tay, Anh Tuan Luu, and Siu Cheung Hui. 2018. Multi-pointer co-attention networks for recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2309–2318.
- Qianqian Wang, Si Li, and Guang Chen. 2018a. Word-driven and context-aware review modeling for recommendation. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 1859–1862.
- Xianchen Wang, Hongtao Liu, Peiyi Wang, Fangzhao Wu, Hongyan Xu, Wenjun Wang, and Xing Xie. 2019. Neural review rating prediction with hierarchical attentions and latent factors. In *International Conference on Database Systems for Advanced Applications*, pages 363–367. Springer.
- Xiang Wang, Xiangnan He, Fuli Feng, Liqiang Nie, and Tat-Seng Chua. 2018b. Tem: Tree-enhanced embedding model for explainable recommendation. In *Proceedings of the 2018 World Wide Web Conference*, pages 1543–1552. International World Wide Web Conferences Steering Committee.
- Xiang Wang, Xiangnan He, Liqiang Nie, and Tat-Seng Chua. 2017. Item silk road: Recommending items from information domains to social users. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 185–194.
- Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google’s neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.
- Alan Zekbik, Tanja Bergmann, and Roland Vollgraf. 2019. Pooled contextualized embeddings for named entity recognition. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 724–728.
- Yongfeng Zhang, Qingyao Ai, Xu Chen, and W Bruce Croft. 2017. Joint representation learning for top-n recommendation with heterogeneous information sources. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1449–1458.
- Yongfeng Zhang and Xu Chen. 2018. Explainable recommendation: A survey and new perspectives. *arXiv preprint arXiv:1804.11192*.
- Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. 2014. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 83–92. ACM.
- Lei Zheng, Vahid Noroozi, and Philip S Yu. 2017. Joint deep modeling of users and items using reviews for recommendation. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 425–434. ACM.

# Transformer-based Approach for Predicting Chemical Compound Structures

**Yutaro Omote**

Ehime University

omote@ai.cs.ehime-u.ac.jp

**Kyoumoto Matsushita**

Fujitsu Laboratories, Ltd.

m.kyoumoto@fujitsu.com

**Tomoya Iwakura**

Fujitsu Laboratories, Ltd.

iwakura.tomoya@fujitsu.com

**Akihiro Tamura**

Doshisha University

aktamura@mail.doshisha.ac.jp

**Takashi Ninomiya**

Ehime University

ninomiya@cs.ehime-u.ac.jp

## Abstract

By predicting chemical compound structures from their names, we can better comprehend chemical compounds written in text and identify the same chemical compound given different notations for database creation. Previous methods have predicted the chemical compound structures from their names and represented them by Simplified Molecular Input Line Entry System (SMILES) strings. However, these methods mainly apply hand-crafted rules, and cannot predict the structures of chemical compound names not covered by the rules. Instead of handcrafted rules, we propose Transformer-based models that predict SMILES strings from chemical compound names. We improve the conventional Transformer-based model by introducing two features: (1) a loss function that constrains the number of atoms of each element in the structure, and (2) a multi-task learning approach that predicts both SMILES strings and InChI strings (another string representation of chemical compound structures). In evaluation experiments, our methods achieved higher F-measures than previous rule-based approaches (Open Parser for Systematic IUPAC Nomenclature and two commercially used products), and the conventional Transformer-based model. We release the dataset used in this paper as a benchmark for the future research<sup>1</sup>.

## 1 Introduction

Knowledge of chemical substances is necessary for developing new materials and drugs, and for synthesizing products from new materials. To utilize such knowledge, researchers have created databases containing the physical property values of chemical substances and the interrelationships among chemical substances.

It is thought that several billions of chemical compounds exist (Lahana, 1999; Hoffmann and

Gastreich, 2019), but only a portion of these are entered into chemical databases. Even PubChem<sup>2</sup>, one of the largest databases of chemical compounds, includes the information of only approximately 100 million chemical compounds. Moreover, databases for chemical domains are manually maintained, which consumes much time and cost. One of the time consuming processes is the integration of the same chemical compounds with different notations. For instance, a chemical structure can be derived from partial structures which are given notational variants, or the notation can fluctuate for a given chemical compound (Watanabe et al., 2019). Therefore, a system that automatically predicts a chemical compound structure from its chemical compound names would improve the database creation procedure.

Structures are most commonly predicted from their notations by rule-based conversion methods (Lowe et al., 2011). Although rule-based conversion can accurately predict the structures of chemical compounds based on systematic nomenclatures such as the International Union of Pure and Applied Chemistry (IUPAC)<sup>3</sup> nomenclature, it often fails the structure prediction of chemical compound names that violate these nomenclatures (e.g., Synonyms<sup>4</sup>).

To improve the low prediction performance of compounds with non-IUPAC names, we propose neural network-based models that predict chemical compound structures represented as Simplified Molecular Input Line Entry System (SMILES) (Weininger, 1988) strings from chemical compound names categorized as Synonyms<sup>5</sup>. In this work, we use the Transformer-based sequence-

<sup>2</sup><https://pubchem.ncbi.nlm.nih.gov/>

<sup>3</sup><https://iupac.org>

<sup>4</sup>PubChem's definition of chemical compound names other than IUPAC names

<sup>5</sup>Our Synonyms excludes DATABASE IDs from the original definition of Synonyms because DATABASE IDs can be efficiently recognized by rules.

<sup>1</sup><http://aiweb.cs.ehime-u.ac.jp/pred-chem-struct>

Name	Type	Name
IUPAC		2-acetyloxybenzoic acid
DATABASE ID (CAS registry number)		50-78-2
ABBREVIATION		ASA
COMMON		aspirin

Table 1: Examples of “aspirin” representations. In this table, ABBREVIATION and COMMON are Synonyms.

to-sequence neural network model (Vaswani et al., 2017) for machine translation, which achieves a state-of-the-art performance in various tasks among the sequence-to-sequence neural network models such as recurrent neural network-based models. To improve the conventional Transformer-based model, we introduce the following two chemical-structure oriented features:

1. A loss function considering the constraints on the number of atoms of each element in the chemical structure.
2. A multi-task learning for predicting both SMILES strings and IUPAC International Chemical Identifier (InChI) (Heller et al., 2015) strings, which are representations for denoting chemical compound structures as strings.

For our experiments, we created a dataset from PubChem for predicting chemical compound structures represented by SMILES strings from Synonyms. The experimental results demonstrate the Transformer-based conversion methods achieve higher F-measures than the existing rule-based methods. In addition, our two proposals (i.e., constraining the number of atoms of each element and multi-task learning of both SMILES strings and InChI strings) improve the performance of the conventional Transformer-based method.

## 2 Preliminary

### 2.1 Chemical Compound Names

In PubChem, the text names of chemical compounds are represented by three main types of notational categories: IUPAC, DATABASE ID, and Synonyms. IUPAC is a systematic nomenclature for chemical compound names. DATABASE ID is the unique identifier of a chemical compound in a database. An example is the Chemical Abstracts Service (CAS)<sup>6</sup> registry number. The Synonyms

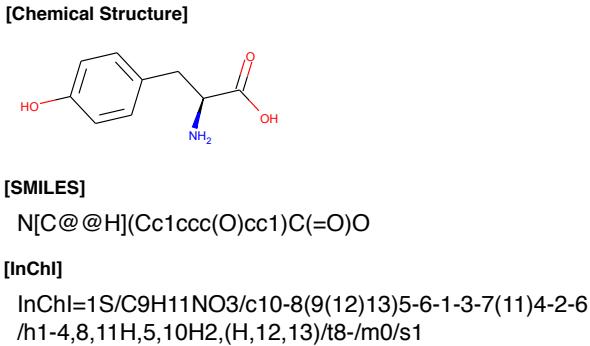


Figure 1: Chemical structure of L-tyrosine (top), and its SMILES (middle) and InChI (bottom) representations

naming category in PubChem includes ABBREVIATION and COMMON. As an example, Table 1 shows various “aspirin” representations.

The IUPAC nomenclature provides a systematic naming under standardized rules, which are easily and accurately converted by rule-based conversion methods (Lowe et al., 2011); (Heller et al., 2015). Provided they are registered in the database, DATABASE IDs are easily converted to their corresponding chemical compounds using dictionary-lookup methods. However, neither rule-based nor dictionary-based approach can convert chemical compound names that are not covered by the rules or dictionaries. Unlike IUPAC and DATABASE ID notations, the naming patterns of Synonyms are complex and widely variable. In many cases, the chemical compound names appearing in documents cannot be converted by rule-based or dictionary-based approaches. Consequently, the prediction performance of chemical compound names is worse in Synonyms than in IUPAC, as shown in section 6.1. In our preliminary experiments, the highest F-measure obtained with an existing tool exceeded 0.96 on IUPAC data, but was reduced to 0.75 on Synonyms data.

<sup>6</sup><https://www.cas.org/>

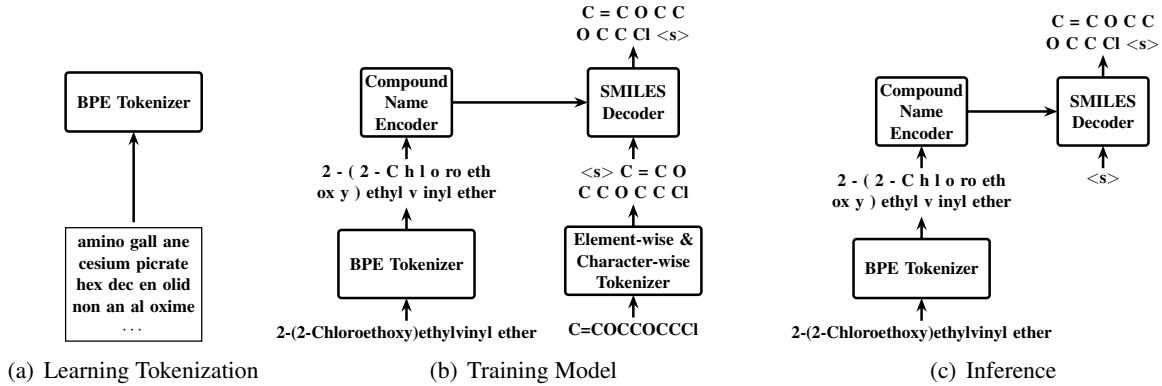


Figure 2: Overview of Transformer-based prediction of SMILES strings from chemical compound names

## 2.2 Representation of Chemical Compound Structures

For multi-task learning, we represented chemical compound structures as SMILES strings and InChI strings. These two representations are major notations of chemical compound structures. We use SMILES strings as the target representation because they are simpler than InChI strings but were sufficiently representative for our purpose (i.e., creating a chemical compound database).

The SMILES (Weininger, 1988) notation system was designed for modern chemical information processing. Based on the principles of molecular graph theory, SMILES allows rigorous structure specification using a very small and natural grammar. SMILES strings are composed of atoms and symbols representing their bonds, branches, rings, and other structural features, assembled into a linear expression of the two-dimensional structure of a molecule. An example of a SMILES string is shown in Figure 1. In this work, we used Canonical SMILES because it uniquely determines the correspondence between chemical structures and SMILES strings.

In the InChI (Heller et al., 2015) representation, the information of a chemical compound structure is represented by five layers. In Figure 1, the layers are separated by “/” symbols. Each layer adds detailed information to the following layer. Because these layers are interrelated, InChI strings are more complex than SMILES strings.

## 3 Proposed Methods

This section presents our proposed methods, namely, our tokenizer training method and sequence-to-sequence models. Let  $\mathcal{X}$  and  $\mathcal{T}$  be a set of chemical compound names and a set

of SMILES strings, respectively. We define a training dataset consisting of  $n$  samples as  $D = \langle(X_1, T_1), \dots, (X_n, T_n)\rangle$ , where  $X_i \in \mathcal{X}$  is a chemical compound name and  $T_i \in \mathcal{T}$  is the SMILES string of  $X_i$  for  $1 \leq i \leq n$ . Our objective is to learn a mapping function  $f$  that realizes  $f(X_i) = T_i$  from  $D$ .

Figure 2 overviews the Transformer-based prediction of SMILES strings from chemical compound names, where  $<\text{s}>$  is a special symbol denoting the start and end of a sequence. Chemical compound names, SMILES, and InChI are long strings without explicit boundaries (such as white spaces in English text). Therefore, to convert chemical compound names to SMILES strings, we propose (a) training of a tokenizer and (b) a Transformer-based approach.

### 3.1 Tokenizer

Chemical compound names can be tokenized by the Open Parser for Systematic IUPAC Nomenclature (OPSIN) (Lowe et al., 2011) tokenizer, a rule-based parser that generates SMILES and InChI strings from chemical compound names (mainly, from IUPAC names). However, some chemical compound names, especially Synonyms, cannot be tokenized by rule-based tokenizers such as OPSIN. In particular, the OPSIN tokenizer is limited to chemical compound names covered by its dictionary and rules; meanwhile (as mentioned above) chemical compound names lack explicit word-boundary markers. To overcome these restrictions, we propose a method that trains tokenizers for Synonyms, SMILES, and InChI representations. Note that InChI is used in a multi-task learning.

To eliminate the unknown tokens, our tokenizer learning method is unsupervised and covers a large

set of chemical compound names. The tokenization is performed by byte pair encoding (BPE) (Sen-nrich et al., 2016)<sup>7</sup>. The BPE-based tokenizer was learned by fastBPE<sup>8</sup>. First, the chemical compound names obtained by the OPSIN tokenizer were segmented because fastBPE requires segmented input text. By virtue of the newly obtained BPE dictionary, the BPE-based tokenizer can tokenize chemical compound names that cannot be handled by the OPSIN tokenizer.

When tokenizing the SMILES strings, each element (e.g., “C”, “O”, “Cl”) identified by regular expressions was regarded as one token. The remaining symbols not covered by regular expressions were divided into single characters, each regarded as one token.

For tokenizing InChI strings, the model was learned on SentencePiece (Kudo and Richardson, 2018), a unigram-based unsupervised training method for word segmentation. Note that InChI strings cannot be tokenized by BPE because the segmentations of InChI strings are not preliminarily given.

### 3.2 Transformer-based Prediction of SMILES Strings from Chemical Compound Names

The Transformer model consists of stacked encoder and decoder layers. Based on self-attention, it attends to tokens in the same sequence, i.e., a single input sequence or a single output sequence. The encoder maps an input sequence to a sequence of vector representations. From this vector representations, the decoder generates an output sequence.

The Transformer-based model predicts SMILES strings from chemical compound names, so its input is a chemical compound name and its output is a SMILES string. During the learning process, the following objective function is minimized:

$$\mathcal{L}_{smiles} = -\log P(T|X; \theta_{enc}, \theta_{smiles}), \quad (1)$$

where  $\theta_{enc}$  and  $\theta_{smiles}$  are the parameter sets of the compound name encoder and SMILES decoder, respectively, and  $X = \langle x_1, x_2, \dots, x_n \rangle$  is the word sequence of a chemical compound name segmented by the BPE model.  $T = \langle t_1, t_2, \dots, t_m \rangle$  is the

<sup>7</sup>In preliminary experiments, BPE achieved a higher F-measure than SentencePiece (Kudo and Richardson, 2018). Therefore, it was used for tokenizing the chemical compound names.

<sup>8</sup><https://github.com/glample/fastBPE>

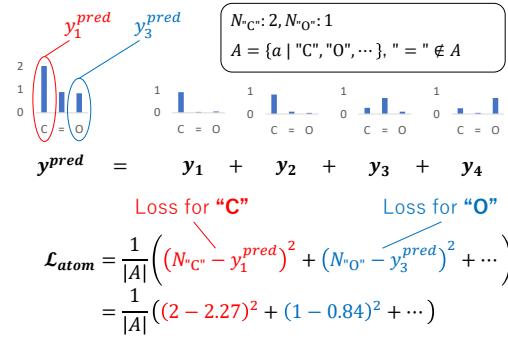


Figure 3: Calculating the constraints on the number of atoms of each element

sequence of elements and symbols in the correct SMILES string of  $X$ .

### 3.3 Training with a Constraint on the Number of Atoms

To correctly predict the chemical structure from a chemical compound name, the number of atoms of each element included in the chemical structure must be fixed. In this subsection, we propose a softmax-based loss function that constrains the number of atoms of each element, that is, we minimize the difference between the numbers of atoms of each element in the predicted and correct SMILES strings. The differences are measured by their squared errors.

The squared errors are computed using the Gumbel softmax (Jang et al., 2016) function, which obtains the probability distribution of the number of atoms of each element in a predicted SMILES string. Let  $\pi_i = (\pi_{i1}, \pi_{i2}, \dots, \pi_{i|\mathcal{V}|})$  be the probability distribution of the  $i$ -th output token from the Transformer model. Then,  $y_i = (y_{i1}, y_{i2}, \dots, y_{i|\mathcal{V}|})$  for the  $i$ -th output token with Gumbel softmax is calculated as follows:

$$y_{ij} = \frac{\exp((\log(\pi_{ij}) + g_{ij})/\tau)}{\sum_{k=1}^{|\mathcal{V}|} \exp((\log(\pi_{ik}) + g_{ik})/\tau)}, \quad (2)$$

$$g_{ij} = -\log(-\log(u_{ij})),$$

$$u_{ij} \sim \text{Uniform}(0, 1),$$

where  $\mathcal{V}$  represents the vocabulary set of SMILES, and  $\tau$  is a hyperparameter of Gumbel softmax. The distribution  $y_i$  approximates an one-hot vector as  $\tau$  decreases, and a uniform distribution as  $\tau$  increases. In this work,  $\tau$  was set to 0.1.

Using Equation 2, the loss function under the

proposed constraints is given by

$$\mathcal{L}_{atom} = \frac{1}{|A|} \sum_{a \in A} (N_a(T) - y_{idx(a)}^{pred})^2, \quad (3)$$

$$\begin{aligned} \mathbf{y}^{pred} &= \mathbf{y}_1 + \mathbf{y}_2 + \cdots + \mathbf{y}_m \\ &= (y_1^{pred}, y_2^{pred}, \dots, y_{|\mathcal{V}|}^{pred}), \end{aligned}$$

where  $A$  is a set of elements,  $N_a(T)$  is a function that returns the number of atoms of element  $a$  in SMILES string  $T$ , and  $idx(a)$  is a function that returns the index of element  $a$  in  $\mathcal{V}$ . Note that  $A$  contains only elemental symbols, and the other features such as symbols representing bonds are absent. More formally, “C”, “O”  $\in A$ , “=”、“#”  $\notin A$ , and  $\mathcal{V} \supset A$ . Each dimension of  $\mathbf{y}^{pred}$  is an estimation of the frequency of the corresponding token of the vocabulary  $\mathcal{V}$  in the predicted SMILES. The proposed constraint calculation uses only the estimation of the elements in  $\mathcal{V}$ . The frequencies of elements not included in the correct SMILES are set to 0.

As an example, Figure 3 shows how the number of atoms of each element is constrained when the correct SMILES string is “CC=O”. As “C” and “O” are elements and “=” is a subsidiary symbol representing a double bond, the proposed constraint function treats the number of atoms of each element (“C” and “O”) as the error to be minimized, and disregards the “=” symbol.

The objective function under the proposed constraints is defined as follows:

$$\mathcal{L}_{smiles} + \lambda_{atom} \mathcal{L}_{atom}, \quad (4)$$

where  $\lambda_{atom}$  is a hyperparameter that controls the degree of considering  $\mathcal{L}_{atom}$ .

### 3.4 Multi-task Learning for Predicting both SMILES Strings and InChI Strings

The same chemical structure is differently represented in a SMILES string and an InChI string. Assuming that the models for predicting SMILES and InChI strings compensate each other, we propose a multi-task learning method that shares the encoder of the name-to-SMILES and name-to-InChI conversion models, and trains both models at the same time.

Let  $\mathcal{I}$  be the set of InChI strings. We define a training dataset consisting of  $n$  samples as  $\hat{D} = \langle (X_1, T_1, I_1), \dots, (X_n, T_n, I_n) \rangle$ , where  $X_i \in \mathcal{X}$ ,  $T_i \in \mathcal{T}$ , and  $I_i \in \mathcal{I}$  for  $1 \leq i \leq n$ . The objective

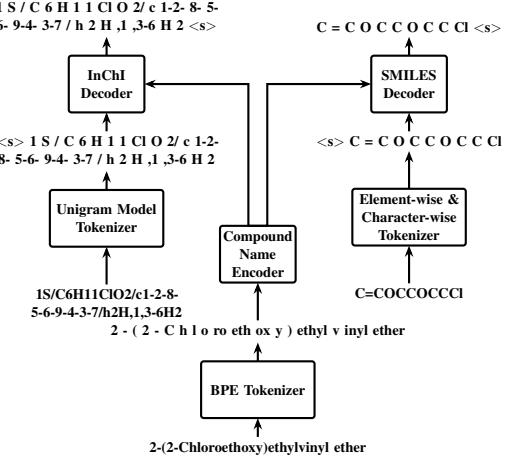


Figure 4: Overview of multi-task learning for predicting both SMILES strings and InChI strings

Split	Size
Training	5,000,000
Development	1,113
Test	11,194

Table 2: Sizes of the training, development, and test datasets

is to learn a function  $\tilde{f}$  from  $\hat{D}$ .  $\tilde{f}(X_i)$  predicts both  $T_i$  and  $I_i$ .

Specifically, the proposed multi-task learning minimizes the following objective function:

$$\begin{aligned} \mathcal{L}_{smiles} + \lambda_{inchi} \mathcal{L}_{inchi}, \quad (5) \\ \mathcal{L}_{inchi} = -\log P(I|X; \theta_{enc}, \theta_{inchi}), \end{aligned}$$

where  $\theta_{inchi}$  and  $\theta_{enc}$  are parameter sets for the InChI decoder and shared encoder, respectively, and  $\lambda_{inchi}$  is a hyperparameter that controls the degree of considering  $\mathcal{L}_{inchi}$ .  $\mathcal{L}_{smiles}$  is calculated by Eq. 1. The method is overviewed in Figure 4.

## 4 Experimental Settings

### 4.1 Data Set

In all experiments, the data comprised a chemical compound name and a correct SMILES string. Using the dump data of PubChem<sup>9</sup> (97M compound records), the chemical compound names were converted to Synonyms associated with each CID<sup>10</sup>, and the correct SMILES strings were converted from isomeric SMILES strings<sup>11</sup> to canon-

<sup>9</sup><ftp://ftp.ncbi.nlm.nih.gov/pubchem/>

<sup>10</sup>PubChem’s compound identifier for a unique chemical structure

<sup>11</sup>SMILES strings written with isotopic and chiral specifications

method		recall	precision	F-measure
Rule-based	OPSIN	0.693	0.836	0.758
	tool A	0.711	0.797	0.752
	tool B	0.653	0.800	0.719
Transformer-based (BPE)	transformer	0.793	0.806	0.799
	atomnum	0.798	0.808	0.803
	inchigen	<b>0.810</b>	0.819	0.814
Transformer-based (OPSIN-TK + BPE)	transformer	0.763	0.873	0.814
	atomnum	0.768	0.876	0.818
	inchigen	0.779	<b>0.886</b>	<b>0.829</b>
Transformer-based (OPSIN-TK)	transformer	0.755	0.868	0.808
	atomnum	0.757	0.867	0.808
	inchigen	0.754	0.869	0.807

Table 3: Evaluation results of each converter for Synonyms. Transformer-based ones are our proposed methods. We evaluated the Transformer-based ones with different three tokenizers, BPE, OPSIN-TK+BPE, and OPSIN-TK.

ical SMILES strings using RDKit<sup>12</sup>. Note that in PubChem, the Synonyms includes the IUPAC names, common names, and IDs of the compounds in chemical compound databases. Here, we used the isomeric SMILES strings because they least overlap with their corresponding CIDs. In the multi-task learning, the InChI strings are also associated with CIDs.

From the dump data, 10,000 CIDs and 100,000 CIDs were randomly selected as the development and test datasets, respectively, and only the two chemical compound names with the longest edit distance were assigned to each CID.

To create Synonyms in the development and test data, chemical compound names like IDs in the chemical compound databases were removed using manually created regular expressions.

In the development and test datasets, duplicate chemical compound names with different CIDs were removed<sup>13</sup>. From the development and test datasets, we removed 820 and 8,241 duplicates, respectively.

As the training dataset, we selected chemical compound names that were categorized as Synonyms that could be tokenized by the OPSIN tokenizer. The size of each dataset is listed in Table 2.

## 4.2 Parameter Settings

The hyperparameters of the Transformer model were set as follows: number of stacks in the encoder and decoder layers = 6, number of heads

= 8, embedding dimension = 512, and dropout probability = 0.1. The loss functions  $\mathcal{L}_{smiles}$  and  $\mathcal{L}_{inchi}$  were computed using a label-smoothing cross entropy with the smoothing parameter  $\epsilon$  set to 0.1. The learning rate was linearly increased to 0.0005 over the first 4,000 steps. In later steps, it was decreased proportionally to the inverse square root of the step number. The optimizer was an Adam (Kingma and Ba, 2015) optimizer with  $\beta_1 = 0.9$ ,  $\beta_2 = 0.98$ , and  $\epsilon = 10^{-8}$ . The model parameters were updated 300,000 times. The hyperparameters  $\lambda_{atom}$  and  $\lambda_{inchi}$  for controlling the degree of constraint consideration were set to 0.7 and 0.3, respectively. The number of merge operations for the BPE-based tokenizer of chemical compound names was set to 500. The vocabulary size for the tokenizer of InChI strings was set to 1,000. We tuned the hyperparameters for our constraints and subword on the development data.

To present the results of our Transformer-based models, we averaged the last 10 checkpoints (saved at 1,000-step intervals) of the Transformer models. We used beam search with a beam size of 4 and length penalty  $\alpha = 0.6$  (Vaswani et al., 2017). The maximum output length of an inference was set to 200.

## 5 Experimental Results

### 5.1 Prediction Performance

The results are shown in Table 3. Here, tool A and tool B are two commercially available tools, atomnum indicates the method based on the number of atoms described in section 3.3, and inchigen denotes the multitask learning method

<sup>12</sup><https://github.com/rdkit/rdkit>

<sup>13</sup>The same chemical compound name may have more than one CID.

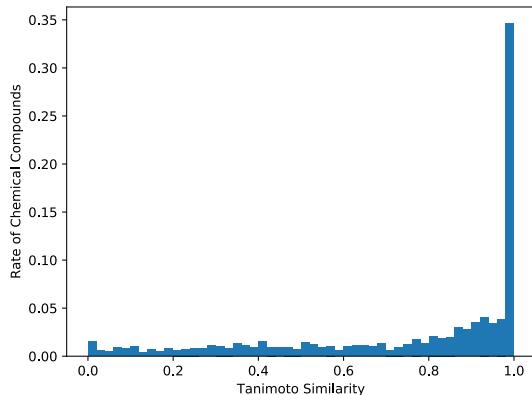


Figure 5: Histogram of Jaccard similarities between incorrect structures generated by `inchigen` with BPE and their correct structures

described in section 3.4. The notations BPE and OPSIN-TK indicate the use of the BPE-based and OPSIN tokenizers, respectively.

As confirmed in Table 3, the proposed methods attained higher prediction performance than the existing rule-based methods and the conventional Transformer-based model. `inchigen` with BPE showed 0.056, 0.062, and 0.095 points higher F-measure than OPSIN, tool A, and tool B, respectively.

The F-measure was further improved by combining the two tokenizers (see the results of OPSIN-TK+BPE in Table 3). In the OPSIN-TK+BPE method, the Transformer-based method with BPE predicted the structures from chemical compound names that could be tokenized by the OPSIN tokenizer. The highest F-measure and precision (0.829 and 0.886, respectively) were achieved by `inchigen` with OPSIN-TK+BPE.

In the Transformer-based models, the OPSIN tokenizer obtained higher precision than the BPE-based tokenizers because approximately 11.5% ( $1,293 / 11,194$ ) of the chemical compounds in the test set could not be tokenized by OPSIN. Consequently, the precision was improved by the reduced number of outputs. In contrast, the recall was lower than in the BPE-based tokenizers.

These results clarify the impact of tokenizer outputs on the recall, precision, and F-measure scores.

## 5.2 Error Analysis

Most of the predictions in the Transformer-based approach were grammatically correct SMILES strings. In this context, “grammatically correct”

means that the chemical structure can be visualized from the predicted SMILES string using RDKit, and does not require the correct SMILES string of a chemical compound name. In particular, `inchigen` with BPE achieved grammatically correct predictions for 99 % of the test data, 10.6–17.4 % higher than OPSIN, tool A, and tool B. To evaluate the usefulness of the Transformer-based approach, we also analyzed the proportion of incorrect structure predictions that were grammatically correct SMILES strings but did not match the correct SMILES strings.

To this end, we measured the Jaccard similarity (Tanimoto similarity)<sup>14</sup> between each structure that was incorrectly predicted by `inchigen` with BPE and the correct structure. The Jaccard similarity, a common technique for measuring chemical compound similarities, is defined as follows:

$$J(X, Y) = \frac{\mathbf{v}_X \cdot \mathbf{v}_Y}{|\mathbf{v}_X + \mathbf{v}_Y| - \mathbf{v}_X \cdot \mathbf{v}_Y},$$

where the  $\mathbf{v}_X$  and  $\mathbf{v}_Y$  are binary chemical fingerprints of chemical compounds X and Y, respectively, represented by binary vectors.  $|\mathbf{v}|$  is the L1 norm of  $\mathbf{v}$ , and  $\mathbf{v}_X \cdot \mathbf{v}_Y$  is the inner product of  $\mathbf{v}_X$  and  $\mathbf{v}_Y$ . Here, a chemical fingerprint expresses a chemical compound structure as a calculable vector. A famous type of fingerprint is a series of binary digits (bits) that represent the presence or absence of particular partial structures in the chemical compound. For example, the Molecular Access System key (Durant et al., 2002), which is used as the fingerprints in the present evaluation, comprises 166 partial structures of chemical compounds. Figure 5 is a histogram of the Jaccard similarity scores obtained in this analysis. We find that most of the incorrect SMILES strings generated by `inchigen` with BPE possessed high Jaccard similarities to the correct SMILES strings. The average Jaccard similarity was 0.753.

An incorrect structure generated by `inchigen` with BPE is compared with its correct structure in Figure 6. The two structures differed only by whether ethylsulfanylbutane or methanethiol was bonded in the partial structures enclosed by the red ellipses. In other words, the two structures are very similar (Jaccard similarity = 0.76).

From this result, we observe that even when the proposed method generates an incorrect structure,

<sup>14</sup>Jaccard similarity, also called the Tanimoto similarity, measures the similarities between pairs of chemical compounds.

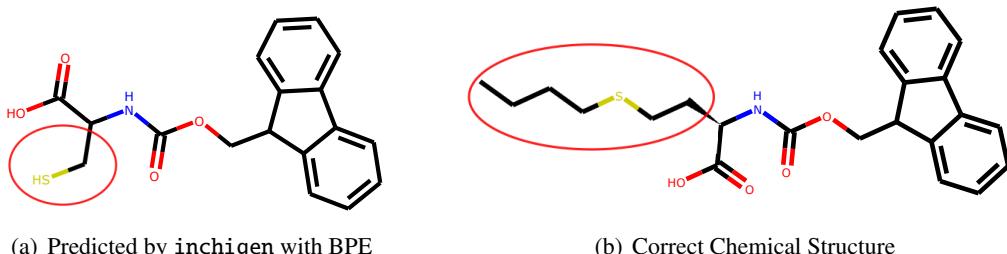


Figure 6: Example of a chemical structure mistakenly for “fmoc-l-buthionine”. The red-edged ellipses enclose the partial structures that differ between the two chemical structures.

the outcome does not deviate greatly from the correct structure.

## 6 Related Work

### 6.1 Predicting SMILES Strings from Chemical Compound Names

OPSIN (Lowe et al., 2011) is a rule-based parser that generates SMILES strings and InChI strings from chemical compound names (mainly from IUPAC names). The OPSIN tokenization approach is based on regular grammar. From a tokenized chemical name, an XML parse tree is constructed. Stepwise operations on this tree are continued until the structure has been reconstructed from the name. The construction is performed on substructures associated with the terms.

As mentioned earlier, many of chemical compound names described in papers and patents do not comply with IUPAC names or other systematic nomenclatures, so are difficult to reconstruct using rule-based methods. In our preliminary experiments using OPSIN and commercially available tools, the F-measures of predicting the IUPAC names in the dataset ranged from 0.878 to 0.960. However, on the Synonyms dataset, the F-measures fell to 0.719-0.758.

### 6.2 Deep Learning methods using SMILES

Recently, SMILES strings have been applied to chemical reaction prediction (Nam and Kim, 2016; Schwaller et al., 2019). The method of Nam and Kim (2016) predicts SMILES strings representing products from SMILES strings representing reactants and reagents. This method employs a sequence-to-sequence model with an attention mechanism based on a recurrent neural network (Bahdanau et al., 2015). Schwaller et al. (2019) achieved higher accuracy than Nam and

Kim (2016)’s model by applying the conventional Transformer model (Vaswani et al., 2017).

Similarly to our study, their models adapt SMILES strings to sequence-to-sequence models, but our target task (predicting chemical structures from their chemical compound names) differs from theirs. To improve the accuracy of our target task, we will improve the update speed and quality of our chemical compounds databases. We also intend to solve other chemistry problems, including chemical reactions, by predictive machine learning.

## 7 Conclusions

This paper introduced our Transformer-based prediction methods, which convert chemical compound names to SMILES strings trained with the constraint of the number of atoms of each element in the SMILES string. We also proposed a multi-task learning approach that simultaneously learns the conversions to SMILES strings and InChI strings. In an experimental comparison evaluation, our proposed method achieved higher F-measures than the existing methods.

In future work, we intend to explore various tokenization methods, and further improve the prediction performance. We also hope to apply the proposed loss function to multi-task learning.

## Acknowledgments

The research results were achieved by the RIKEN AIP-FUJITSU Collaboration Center, Japan.

## References

- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

- Joseph L. Durant, Burton A. Leland, Douglas R. Henry, and James G. Nourse. 2002. [Reoptimization of mdl keys for use in drug discovery](#). *Journal of Chemical Information and Computer Sciences*, 42(6):1273–1280.
- Stephen R Heller, Alan McNaught, Igor Pletnev, Stephen Stein, and Dmitrii Tchekhovskoi. 2015. Inchi, the iupac international chemical identifier. *Journal of cheminformatics*, 7(1):23.
- Torsten Hoffmann and Marcus Gastreich. 2019. [The next level in chemical space navigation: going far beyond enumerable compound libraries](#). *Drug Discovery Today*, 24(5):1148 – 1156.
- Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *ArXiv*, abs/1611.01144.
- Diederik P. Kingma and Jimmy Ba. 2015. [Adam: A method for stochastic optimization](#). In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.
- Taku Kudo and John Richardson. 2018. [SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing](#). In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.
- Roger Lahana. 1999. [How many leads from hts?](#) *Drug Discovery Today*, 4(10):447 – 448.
- Daniel M. Lowe, Peter T. Corbett, Peter Murray-Rust, and Robert C. Glen. 2011. Chemical name to structure: Opsin, an open source solution. *Journal of Chemical Information and Modeling*, 51(3):739–753.
- Juno Nam and Jurae Kim. 2016. [Linking the neural machine translation and the prediction of organic chemistry reactions](#).
- Philippe Schwaller, Teodoro Laino, Théophile Gaudin, Peter Bolgar, Christopher A Hunter, Costas Bekas, and Alpha A Lee. 2019. Molecular transformer: A model for uncertainty-calibrated chemical reaction prediction. *ACS central science*, 5(9):1572–1583.
- Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. [Neural machine translation of rare words with subword units](#). In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. [Attention is all you need](#). In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.
- Taiki Watanabe, Akihiro Tamura, Takashi Ninomiya, Takuya Makino, and Tomoya Iwakura. 2019. Multi-task learning for chemical named entity recognition with chemical compound paraphrasing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6243–6248.
- David Weininger. 1988. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of Chemical Information and Computer Sciences*, 28(1):31–36.

# Chinese Grammatical Correction Using BERT-based Pre-trained Model

Hongfei Wang, Michiki Kurosawa\*, Satoru Katsumata<sup>†</sup> and Mamoru Komachi

Tokyo Metropolitan University

wang-hongfei@ed.tmu.ac.jp, m-kurosawa@nri.co.jp

satoru.katsumata@retrieva.jp, komachi@tmu.ac.jp

## Abstract

In recent years, pre-trained models have been extensively studied, and several downstream tasks have benefited from their utilization. In this study, we verify the effectiveness of two methods that incorporate a BERT-based pre-trained model developed by Cui et al. (2020) into an encoder-decoder model on Chinese grammatical error correction tasks. We also analyze the error type and conclude that sentence-level errors are yet to be addressed.

## 1 Introduction

Grammatical error correction (GEC) can be regarded as a sequence-to-sequence task. GEC systems receive an erroneous sentence written by a language learner and output the corrected sentence. In previous studies that adopted neural models for Chinese GEC (Ren et al., 2018; Zhou et al., 2018), the performance was improved by initializing the models with a distributed word representation, such as Word2Vec (Mikolov et al., 2013). However, in these methods, only the embedding layer of a pre-trained model was used to initialize the models.

In recent years, pre-trained models based on Bidirectional Encoder Representations from Transformers (BERT) have been studied extensively (Devlin et al., 2019; Liu et al., 2019), and the performance of many downstream Natural Language Processing (NLP) tasks has been dramatically improved by utilizing these pre-trained models. To learn existing knowledge of a language, a BERT-based pre-trained model is trained on a large-scale corpus using the encoder of Transformer (Vaswani et al., 2017). Subsequently, for a downstream task, a neural network model is initialized with the weights learned by a pre-trained model that has the same structure and is fine-tuned on training data of

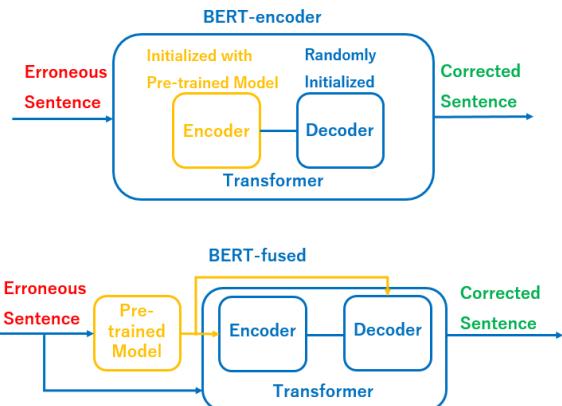


Figure 1: Two methods for incorporating a pre-trained model into the GEC model.

the downstream task. Using this two-stage method, the performance is expected to improve because downstream tasks are informed by the knowledge learned by the pre-trained model.

Recent works (Kaneko et al., 2020; Kantor et al., 2019) show that BERT helps improve the performance on the English GEC task. As the Chinese pre-trained models are developed and released continuously (Cui et al., 2020; Zhang et al., 2019), the Chinese GEC task may also benefit from using those pre-trained models.

In this study, as shown in Figure 1, we develop a Chinese GEC model based on Transformer with a pre-trained model using two methods: first, by initializing the encoder with the pre-trained model (BERT-encoder); second, by utilizing the technique proposed by Zhu et al. (2020), which uses the pre-trained model for additional features (BERT-fused); on the Natural Language Processing and Chinese Computing (NLPCC) 2018 Grammatical Error Correction shared task test dataset (Zhao et al., 2018), our single models obtain  $F_{0.5}$  scores of 29.76 and 29.94 respectively, which is similar to the performance of ensemble models developed

\* Currently at Nomura Research Institute, Ltd.

<sup>†</sup> Currently at Retrieva, Inc.

by the top team of the shared task. Moreover, using a 4-ensemble model, we obtain an  $F_{0.5}$  score of 35.51, which outperforms the results from the top team by a large margin. We annotate the error types of the development data; the results show that word-level errors dominate all error types and that sentence-level errors remain challenging and require a stronger approach.

## 2 Related Work

Given the success of the shared tasks on English GEC at the Conference on Natural Language Learning (CoNLL) (Ng et al., 2013, 2014), a Chinese GEC shared task was performed at the NLPCC 2018. In this task, approximately one million sentences from the language learning website Lang-8<sup>1</sup> were used as training data and two thousand sentences from the PKU Chinese Learner Corpus (Zhao et al., 2018) were used as test data. Here, we briefly describe the three methods with the highest performance.

First, Fu et al. (2018) combined a 5-gram language model-based spell checker with subword-level and character-level encoder-decoder models using Transformer to obtain five types of outputs. Then, they re-ranked these outputs using the language model. Although they reported a high performance, several models were required, and the combination method was complex.

Second, Ren et al. (2018) utilized a convolutional neural network (CNN), such as in Chollampatt and Ng (2018). However, because the structure of the CNN is different from that of BERT, it cannot be initialized with the weights learned by the BERT.

Last, Zhao and Wang (2020) proposed a dynamic masking method that replaces the tokens in the source sentences of the training data with other tokens (e.g. [PAD] token). They achieved state-of-the-art results on the NLPCC 2018 Grammar Error Correction shared task without using any extra knowledge. This is a data augmentation method that can be a supplement for our study.

## 3 Methods

In the proposed method, we construct a correction model using Transformer, and incorporate a Chinese pre-trained model developed by Cui et al. (2020) in two ways as described in the following sections.

<sup>1</sup><https://lang-8.com/>

### 3.1 Chinese Pre-trained Model

We use a BERT-based model as our pre-trained model. BERT is mainly trained with a task called Masked Language Model. In the Masked Language Model task, some tokens in a sentence are replaced with masked tokens ([MASK]) and the model has to predict the replaced tokens.

In this study, we use the Chinese-RoBERTa-wwm-ext model developed by Cui et al. (2020). The main difference between Chinese-RoBERTa-wwm-ext and the original BERT is that the latter uses whole word masking (WWM) to train the model. In WWM, when a Chinese character is masked, other Chinese characters that belong to the same word should also be masked.

### 3.2 Grammatical Error Correction Model

In this study, we use Transformer as the correction model. Transformer has shown excellent performance in sequence-to-sequence tasks, such as machine translation, and has been widely adopted in recent studies on English GEC (Kiyono et al., 2019; Junczys-Dowmunt et al., 2018).

However, a BERT-based pre-trained model only uses the encoder of Transformer; therefore, it cannot be directly applied to sequence-to-sequence tasks that require both an encoder and a decoder, such as GEC. Hence, we incorporate the encoder-decoder model with the pre-trained model in two ways as described in the following subsections.

**BERT-encoder** We initialize the encoder of Transformer with the parameters learned by Chinese-RoBERTa-wwm-ext; the decoder is initialized randomly. Finally, we fine-tune the initialized model on Chinese GEC data.

**BERT-fused** Zhu et al. (2020) proposed a method that uses a pre-trained model as the additional features. In this method, input sentences are fed into the pre-trained model and representations from the last layer of the pre-trained model are acquired first. Then, the representations will interact with the encoder and decoder by using attention mechanism. Kaneko et al. (2020) verified the effectiveness of this method on English GEC tasks.

## 4 Experiments

### 4.1 Experimental Settings

**Data** In this study, we use the data provided by the NLPCC 2018 Grammatical Error Correction

shared task. We first segment all sentences into characters because the Chinese pre-trained model we used is character-based. In the GEC task, source and target sentences do not tend to change significantly. Considering this, we filter the training data by excluding sentence pairs that meet the following criteria: i) the source sentence is identical to the target sentence; ii) the edit distance between the source sentence and the target sentence is greater than 15; iii) the number of characters of the source sentence or the target sentence exceeds 64. Once the training data were filtered, we obtained 971,318 sentence pairs.

Because the NLPCC 2018 Grammatical Error Correction shared task did not provide development data, we opted to randomly extract 5,000 sentences from the training data as the development data following [Ren et al. \(2018\)](#).

The test data consist of 2,000 sentences extracted from the PKU Chinese Learner Corpus. According to [Zhao et al. \(2018\)](#), the annotation guidelines follow the minimum edit distance principle ([Nagata and Sakaguchi, 2016](#)), which selects the edit operation that minimizes the edit distance from the original sentence.

**Model** We implement the Transformer model using fairseq 0.8.0.<sup>2</sup> and load the pre-trained model using pytorch\_transformer 2.2.0.<sup>3</sup>

We then train the following models based on Transformer.

**Baseline:** a plain Transformer model that is initialized randomly without using a pre-trained model.

**BERT-encoder:** the correction model introduced in Section 3.2.

**BERT-fused:** the correction model introduced in Section 3.2. We use the implementation provided by [Zhu et al. \(2020\)](#).<sup>4</sup>

Finally, we train a 4-ensemble BERT-encoder model and a 4-ensemble BERT-fused model.

More details on the training are provided in the appendix A.

**Evaluation** As the evaluation is performed on word-unit, we strip all delimiters from the system output sentences and segment the sentences using

<sup>2</sup><https://github.com/pytorch/fairseq>

<sup>3</sup><https://github.com/huggingface/transformers>

<sup>4</sup><https://github.com/bert-nmt/bert-nmt>

[Our models]	<b>P</b>	<b>R</b>	<b>F<sub>0.5</sub></b>
Baseline	25.14	14.34	21.85
BERT-encoder	32.67	22.19	29.76
BERT-fused	32.11	<b>23.57</b>	29.94
BERT-encoder (4-ensemble)	41.94	22.02	35.51
BERT-fused (4-ensemble)	32.20	23.16	29.87
<hr/>			
[SOTA Result]			
<a href="#">Zhao and Wang (2020)</a>	44.36	22.18	<b>36.97</b>
<hr/>			
[NLPCC 2018]			
<a href="#">Fu et al. (2018)</a>	35.24	18.64	29.91
<a href="#">Ren et al. (2018)</a>	41.73	13.08	29.02
<a href="#">Ren et al. (2018) (4-ensemble)</a>	<b>47.63</b>	12.56	30.57

Table 1: Experimental results on the NLPCC 2018 Grammatical Error Correction shared task.

the pkunlp<sup>5</sup> provided in the NLPCC 2018 Grammatical Error Correction shared task.

Based on the setup of the NLPCC 2018 Grammatical Error Correction shared task, the evaluation is conducted using *MaxMatch* (M2).<sup>6</sup>

## 4.2 Evaluation Results

Table 1 summarizes the experimental results of our models. We run the single models four times, and report the average score. For comparison, we also cite the result of the state-of-the-art model ([Zhao and Wang, 2020](#)) and the results of the models developed by two teams in the NLPCC 2018 Grammatical Error Correction shared task.

The performances of BERT-encoder and BERT-fused are significantly superior to that of the baseline model and are comparable to those achieved by the two teams in the NLPCC 2018 Grammatical Error Correction shared task, indicating the effectiveness of adopting the pre-trained model.

The BERT-encoder (4-ensemble) model yields an F<sub>0.5</sub> score nearly 5 points higher than the highest-performance model in the NLPCC 2018 Grammatical Error Correction shared task. However, there is no improvement for the BERT-fused (4-ensemble) model compared with the single BERT-fused model. We find that the performance of the BERT-fused model depends on the warm-up model. Compared with [Kaneko et al. \(2020\)](#) using a state-of-the-art model to warm-up their BERT-fused model, we did not use a warm-up model in this work. The performance noticeably drops when we try to warm-up the BERT-fused model from a weak baseline model, therefore, the BERT-fused model may perform better when warmed-up from a

<sup>5</sup><http://59.108.48.12/lcwm/pkunlp/downloads/libgrass-ui.tar.gz>

<sup>6</sup><https://github.com/nusnlp/m2scorer>

src	持别是北京，没有“自然”的感觉。	人们在一辈子经验很多事情。
gold	特別是北京，没有“自然”的感觉。	人们在一辈子经历很多事情。
baseline	持別是北京，没有“自然”的感觉。	人们在一輩子经历了很多事情。
BERT-encoder	特別是北京，没有“自然”的感觉。	人们一輩子会经历很多事情。
Translation	Especially in Beijing, there is no <i>natural</i> feeling.	People <b>experience</b> many things in their lifetime.

Table 2: Source sentence, gold edit, and output of our models.

Error Type	Number of errors	Examples
B	9	最后，要 <u>关注</u> {关注}一些关于天气预报的新闻。 (Finally, pay attention to some weather forecast news.)
CC	35	有一天晚上他下了 <u>决定</u> {决心}向富丽堂皇的宫殿里走，偷偷的{地}进入宫内。 (One night he decided to walk to the magnificent palace, and sneaked in it secretly.)
CQ	30	在上海我总是住 <u>NONE</u> {在}一家特定 <u>NONE</u> {的}酒店。 (I always stay in the same hotel in Shanghai.)
CD	21	我很喜欢 <u>念</u> {NONE}读小说。 (I like to read novels.)
CJ	35	……但是同时也对环境问题 <u>NONE</u> 日益严重造成了{造成了日益严重的}空气污染问题。 (But on the meanwhile, it also aggravated the problem of air pollution.)

Table 3: Examples of each error type. The underlined tokens are detected errors that should be replaced with the tokens in braces.

Type	Detection			Correction		
	P	R	F <sub>0.5</sub>	P	R	F <sub>0.5</sub>
<b>BERT-encoder</b>						
B	<b>80.0</b>	<b>55.6</b>	<b>73.5</b>	<b>80.0</b>	<b>55.6</b>	<b>73.5</b>
CC	62.5	31.4	52.2	43.8	20.0	35.4
CQ	65.0	43.3	59.1	45.0	30.0	40.9
CD	58.3	28.6	48.3	50.0	28.6	43.5
CJ	56.5	42.9	53.1	4.3	2.9	3.9
<b>BERT-fused</b>						
B	<b>80.0</b>	44.4	<b>69.0</b>	<b>80.0</b>	44.4	<b>69.0</b>
CC	61.9	42.9	56.9	38.1	22.9	33.6
CQ	69.0	<b>63.3</b>	67.8	44.8	<b>46.7</b>	45.2
CD	71.4	42.9	63.0	57.1	38.1	51.9
CJ	63.2	34.3	54.1	15.8	8.6	13.5

Table 4: Detection and correction performance of BERT-encoder and BERT-fused models on each type of error.

stronger model (e.g., the model proposed by Zhao and Wang (2020)).

For the state-of-the-art result achieved by Zhao and Wang (2020), both the precision and the recall are comparatively high, and they therefore obtain the best F<sub>0.5</sub> score.

Additionally, the precision of the models that used a pre-trained model is lower than that of the models proposed by the two teams; conversely, the recall is significantly higher.

## 5 Discussion

**Case Analysis** Table 2 shows the sample outputs.

In the first example, the spelling error 特別 is accurately corrected to 特別 (which means *especially*) by the proposed model, whereas it is not corrected by the baseline model. Hence, it appears

that the proposed model captures context more efficiently by using the pre-trained model through the WWM strategy.

In the second example, the output of the proposed model is more fluent, although the correction made by the proposed model is different from the gold edit. The proposed model not only changed the wrong word 经验 (which usually means the noun *experience*) to 经历 (which usually means the verb *experience*), but also added a new word 会 (*would, could*); this addition makes the sentence more fluent. It appears that the proposed model can implement additional changes to the source sentence because the pre-trained model is trained with a large-scale corpus. However, this type of change may affect the precision because the gold edit in this dataset followed the principle of minimum edit distance (Zhao et al., 2018).

**Error Type Analysis** To understand the error distribution of Chinese GEC, we annotate 100 sentences of development data and obtain 130 errors (one sentence may contain more than one error). We refer to the annotation of the HSK learner corpus<sup>7</sup> and adopt five categories of error: B, CC, CQ, CD, and CJ. B denotes character-level errors, which are mainly spelling and punctuation errors. CC, CQ, and CD are word-level errors, which are word selection, missed word, and redundant word errors, respectively. CJ denotes sentence-level errors which contain several complex errors, such as word order and lack of subject errors. Several

<sup>7</sup><http://hsk.blcu.edu.cn/>

examples are presented in Table 3. Based on the number of errors, it is evident that word-level errors (CC, CQ, and CD) are the most frequent.

Table 4 lists the detection and correction results of the BERT-encoder and BERT-fused models for each error type. The two models perform poorly on sentence-level errors (CJ), which often involve sentence reconstructions, demonstrating that this is a difficult task. For character-level errors (B), the models achieve better performance than for other error types. Compared with the correction performance, the systems indicate moderate detection performance, demonstrating that the systems address error positions appropriately. With respect to the difference in performance of the two systems on each error type, we can conclude that BERT-encoder performs better on character-level errors (B), and BERT-fused performs better on other error types.

## 6 Conclusion

In this study, we incorporated a pre-trained model into an encoder-decoder model using two methods on Chinese GEC tasks. The experimental results demonstrate the usefulness of the BERT-based pre-trained model in the Chinese GEC task. Additionally, our error type analysis showed that sentence-level errors remain to be addressed.

## Acknowledgments

This work has been partly supported by the programs of the Grant-in-Aid for Scientific Research from the Japan Society for the Promotion of Science (JSPS KAKENHI) Grant Numbers 19K12099 and 19KK0286.

## References

- Shamil Chollampatt and Hwee Tou Ng. 2018. A multi-layer convolutional encoder-decoder neural network for grammatical error correction. In *AAAI*.
- Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Shijin Wang, and Guoping Hu. 2020. Revisiting pre-trained models for Chinese natural language processing. In *Findings of EMNLP*.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.
- Kai Fu, Jun Huang, and Yitao Duan. 2018. Youdao’s winning solution to the NLPCC-2018 task 2 chal-
- lenge: A neural machine translation approach to Chinese grammatical error correction. In *NLPCC*.
- Marcin Junczys-Dowmunt, Roman Grundkiewicz, Shubha Guha, and Kenneth Heafield. 2018. Approaching neural grammatical error correction as a low-resource machine translation task. In *NAACL-HLT*.
- Masahiro Kaneko, Masato Mita, Shun Kiyono, Jun Suzuki, and Kentaro Inui. 2020. Encoder-decoder models can benefit from pre-trained masked language models in grammatical error correction. In *ACL*.
- Yoav Kantor, Yoav Katz, Leshem Choshen, Edo Cohen-Karlik, Naftali Liberman, Assaf Toledo, Amir Menczel, and Noam Slonim. 2019. Learning to combine grammatical error corrections. In *BEA@ACL*.
- Shun Kiyono, Jun Suzuki, Masato Mita, Tomoya Mizumoto, and Kentaro Inui. 2019. An empirical study of incorporating pseudo data into grammatical error correction. In *EMNLP-IJCNLP*.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke S. Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *ArXiv*.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*.
- Ryo Nagata and Keisuke Sakaguchi. 2016. Phrase structure annotation and parsing for learner English. In *ACL*.
- Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. 2014. The CoNLL-2014 shared task on grammatical error correction. In *CoNLL*.
- Hwee Tou Ng, Siew Mei Wu, Yuanbin Wu, Christian Hadiwinoto, and Joel Tetreault. 2013. The CoNLL-2013 shared task on grammatical error correction. In *CoNLL*.
- Hongkai Ren, Liner Yang, and Endong Xun. 2018. A sequence to sequence learning for Chinese grammatical error correction. In *NLPCC*.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.
- Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced language representation with informative entities. In *ACL*.
- Yuanyuan Zhao, Nan Jiang, Weiwei Sun, and Xiaojun Wan. 2018. Overview of the NLPCC 2018 shared task: Grammatical error correction. In *NLPCC*.

Zewei Zhao and Houfeng Wang. 2020. MaskGEC: Improving neural grammatical error correction via dynamic masking. In *AAAI*.

Junpei Zhou, Chen Li, Hengyou Liu, Zuyi Bao, Guangwei Xu, and Linlin Li. 2018. Chinese grammatical error correction using statistical and neural models. In *NLPCC*.

Jinhua Zhu, Yingce Xia, Lijun Wu, Di He, Tao Qin, Wengang Zhou, Houqiang Li, and Tieyan Liu. 2020. Incorporating BERT into neural machine translation. In *ICLR*.

## A Appendices

Table 5 shows the training details for each model.

<b>Baseline</b>	
Architecture	Encoder (12-layer), Decoder (12-layer)
Learning rate	$1 \times 10^{-5}$
Batch size	32
Optimizer	Adam ( $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 1 \times 10^{-8}$ )
Max epochs	20
Loss function	cross-entropy
Dropout	0.1
<b>BERT-encoder</b>	
Architecture	Encoder (12-layer), Decoder (12-layer)
Learning rate	$3 \times 10^{-5}$
Batch size	32
Optimizer	Adam ( $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 1 \times 10^{-8}$ )
Max epochs	20
Loss function	cross-entropy
Dropout	0.1
<b>BERT-fused</b>	
Architecture	Transformer (big)
Learning rate	$3 \times 10^{-5}$
Batch size	32
Optimizer	Adam ( $\beta_1 = 0.9, \beta_2 = 0.98, \epsilon = 1 \times 10^{-8}$ )
Max epochs	20
Loss function	label smoothed cross-entropy ( $\epsilon_{ls} = 0.1$ )
Dropout	0.3

Table 5: Training details for each model.

# Neural Gibbs Sampling for Joint Event Argument Extraction

Xiaozhi Wang<sup>1\*</sup>, Shengyu Jia<sup>3\*</sup>, Xu Han<sup>1</sup>, Zhiyuan Liu<sup>1,2†</sup>,  
Juanzi Li<sup>1,2</sup>, Peng Li<sup>4</sup>, Jie Zhou<sup>4</sup>

<sup>1</sup>Department of Computer Science and Technology, BNRist;

<sup>2</sup>KIRC, Institute for Artificial Intelligence;

<sup>3</sup>Department of Electrical Engineering,

Tsinghua University, Beijing, 100084, China

<sup>4</sup>Pattern Recognition Center, WeChat AI, Tencent Inc, China

{wangxz20, jsy20, hanxu17}@mails.tsinghua.edu.cn

## Abstract

Event Argument Extraction (EAE) aims at predicting event argument roles of entities in text, which is a crucial subtask and bottleneck of event extraction. Existing EAE methods either extract each event argument roles independently or sequentially, which cannot adequately model the joint probability distribution among event arguments and their roles. In this paper, we propose a Bayesian model named Neural Gibbs Sampling (NGS) to jointly extract event arguments. Specifically, we train two neural networks to model the prior distribution and conditional distribution over event arguments respectively and then use Gibbs sampling to approximate the joint distribution with the learned distributions. For overcoming the shortcoming of the high complexity of the original Gibbs sampling algorithm, we further apply simulated annealing to efficiently estimate the joint probability distribution over event arguments and make predictions. We conduct experiments on the two widely-used benchmark datasets ACE 2005 and TAC KBP 2016. The Experimental results show that our NGS model can achieve comparable results to existing state-of-the-art EAE methods. The source code can be obtained from <https://github.com/THU-KEG/NGS>.

## 1 Introduction

Event argument extraction (EAE) is a crucial subtask of Event Extraction, which aims at predicting entities and their event argument roles in event mentions. For instance, given the sentence “Fox’s stock price rises after the acquisition of its entertainment businesses by Disney”, the event detection (ED) model will first identify the trigger word “acquisition” triggering a Transfer-Ownership event. Then, with the trigger word and event type,

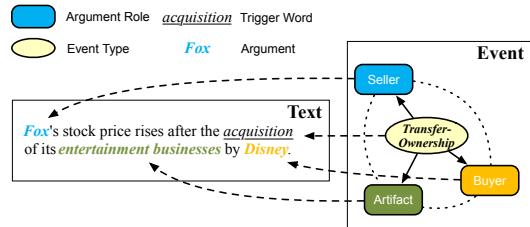


Figure 1: An example of event extraction, including event detection and event argument extraction.

the EAE model is required to identify that “Fox” and “Disney” are event arguments whose roles are “Seller” and “Buyer” respectively. As ED is well-studied in recent years (Liu et al., 2018a; Nguyen and Grishman, 2018; Zhao et al., 2018; Wang et al., 2019a), EAE becomes the bottleneck and has drawn growing attention.

As EAE is the bottleneck of event extraction, especially is also important for various NLP applications (Yang et al., 2003; Basile et al., 2014; Cheng and Erk, 2018), intensive efforts have already been devoted to designing effective EAE systems. The early feature-based methods (Patwardhan and Riloff, 2009; Gupta and Ji, 2009) manually design sophisticated features and heuristic rules to extract event arguments. As the development of neural networks, various neural methods adopt convolutional (Chen et al., 2015) or recurrent (Nguyen et al., 2016) neural networks to automatically represent sentence semantics with low-dimensional vectors, and independently determine argument roles with the vectors. Recently, some advanced techniques have also been adopted to further enhance the performance of EAE models, such as zero-shot learning (Huang et al., 2018), multi-modal integration (Zhang et al., 2017) and weak supervision (Chen et al., 2017).

However, above-mentioned methods do not model the correlation among event arguments in

\* indicates equal contribution

† Corresponding author: Z.Liu (liuzy@tsinghua.edu.cn)

event mentions. As shown in Figure 1, all event arguments are correlated with each other. It is more likely to see a “Seller” when you have seen a “Buyer” and an “Artifact” in event mentions, and vice versa. Formally, with  $x_i$  denoting the random variable of the  $i$ -th event argument candidate, the required probability distribution for EAE is  $P(x_1, x_2, \dots, x_n|o)$ , where  $o$  is the observation from sentence semantics of event mentions. The existing methods which independently extract event arguments solely model  $P(x_i|o)$ , totally ignoring the correlation among event arguments, which may lead models to trapping in a local optimum.

Recently, some proactive works view EAE as a sequence labeling problem (Yang and Mitchell, 2016; Nguyen et al., 2016; Zeng et al., 2018) and adopt conditional random field (CRF) with the Viterbi algorithm (Rabiner, 1989) to solve the problem. These explorations consider the correlation of event arguments unintentionally. Yet limited by the Markov property, their linear-chain CRF only considers the correlation between two adjacent event arguments in the sequence and finds a maximum likelihood path to model the joint distribution, i.e., these sequence models cannot adequately handle the complex situation that each event argument is correlated with each other in event mentions, just like the example shown in Figure 1.

To adequately model the genuine joint distribution  $P(x_1, x_2, \dots, x_n|o)$  rather than  $\prod_i^n P(x_i|o)$  for EAE, we propose a Bayesian method named **Neural Gibbs Sampling (NGS)** inspired by previous work (Finkel et al., 2005; Sun et al., 2014). Gibbs sampling (Geman and Geman, 1987) is a Markov Chain Monte Carlo (MCMC) algorithm, which defines a Markov chain in the space of possible variable assignments whose stationary distribution is the desired joint distribution. Then, a Monte Carlo method is adopted to sample a sequence of observations, and the sampled sequence can be used to approximate the joint distribution.

More specifically, for NGS, we first adopt a neural network to model the prior distribution  $P_p(x_i|o)$  and independently predict an argument role for each event argument candidate to get an initial state for the random variable sequence  $x_1, x_2, \dots, x_n$ , which is similar to the previous methods. Then, we train a special neural network to model the conditional probability distribution  $P_c(x_i|x_1, x_2, \dots, x_{i-1}, x_{i+1}, \dots, x_n, o)$  and iteratively change the sequence state by this conditional

distribution. Intuitively, the network modeling the conditional probability distribution aims to predict unknown argument roles based on both sentence semantics and some known argument roles. After enough steps, the state of the sequence will accurately follow the posterior joint distribution  $P(x_1, x_2, \dots, x_n|o)$ , and the most frequent state in history will be the best result of EAE.

Considering that it will take many steps to accurately estimate the shape of the joint distribution and each step uses neural networks for inference, it is time-consuming and impractical. Due to what we want for EAE is the max-likelihood state of the argument roles, we follow Geman and Geman (1987) and adopt **simulated annealing** (Kirkpatrick et al., 1983) to efficiently find the max-likelihood state based on the Gibbs sampling.

To conclude, our main contributions can be summarized as follows:

(1) Our NGS method combines both the advantages of neural networks and the Gibbs sampling method. The neural networks have shown their strong ability to fit a distribution from data. Gibbs sampling has remarkable advantages in performing Bayesian inference and modeling the complex correlation among event arguments.

(2) Considering the shortcoming of high complexity of the original Gibbs sampling algorithm, we further apply simulated annealing to efficiently estimate the joint probability distribution and find the max-likelihood state for NGS.

(3) Experimental results on the widely-used benchmark datasets ACE 2005 and TAC KBP 2016 show that our NGS works well to consider the correlation among event arguments and achieves the state-of-the-art results. The experiments also show that the simulated annealing method can significantly improve the convergence speed and the stability of Gibbs sampling, which demonstrate that our NGS is both effective and efficient.

## 2 Related Work

Event Extraction (EE) aims to extract structured information from plain text, which is a challenging task in the field of information extraction. EE consists of two subtasks, one is event detection (ED) to detect words triggering events and identify event types, the other is event argument extraction (EAE) to extract argument entities in event mentions and identify event argument roles. As EE is important and beneficial for various downstream

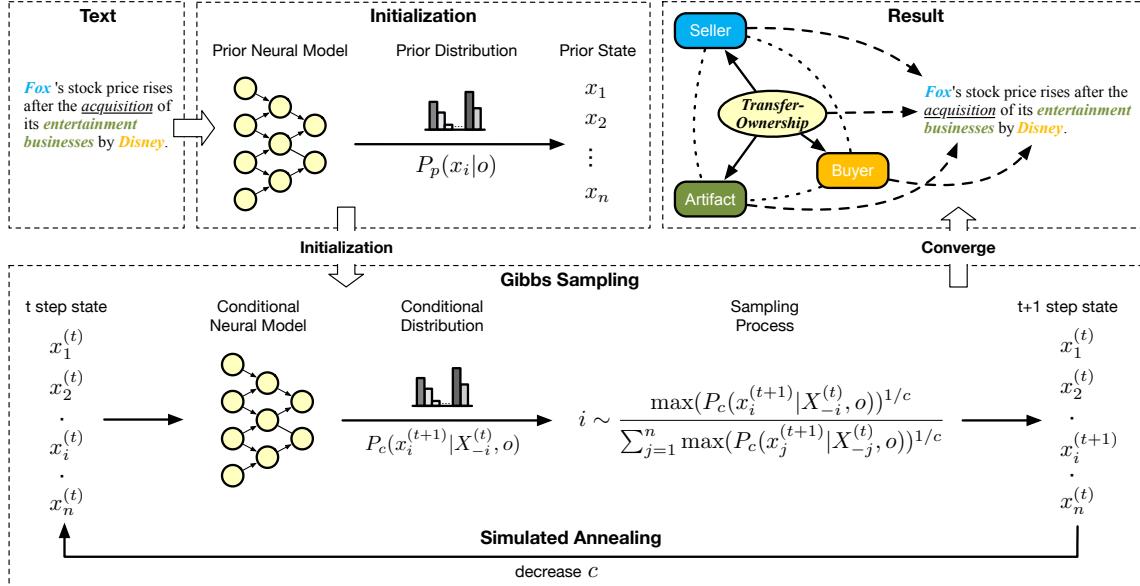


Figure 2: Overall framework of our Neural Gibbs Sampling model.

NLP tasks, e.g., question answering (Yang et al., 2003), information retrieval (Basile et al., 2014), and reading comprehension (Cheng and Erk, 2018), it has attracted wide attentions recently.

ED has been well-studied by the previous works due to its simple and clear definition, including feature-based and rule-based methods (Ahn, 2006; Ji and Grishman, 2008; Gupta and Ji, 2009; Riedel et al., 2010; Hong et al., 2011; McClosky et al., 2011; Huang and Riloff, 2012a,b; Araki and Mita-mura, 2015; Li et al., 2013; Yang and Mitchell, 2016; Liu et al., 2016b), neural methods (Chen et al., 2015; Nguyen and Grishman, 2015; Nguyen et al., 2016; Duan et al., 2017; Nguyen et al., 2016; Ghaeini et al., 2016; Lin et al., 2018), the methods with external heterogeneous knowledge (Liu et al., 2016a, 2017; Zhang et al., 2017; Duan et al., 2017; Zhao et al., 2018; Liu et al., 2018b). Some advanced architectures, such as graph convolutional networks (Nguyen and Grishman, 2018) and adversarial training (Hong et al., 2018; Wang et al., 2019a), have also been applied recently.

As ED models has achieved relatively promising results, the more difficult EAE becomes the bottleneck of EE, and have drawn growing research interests. The early works (Patwardhan and Riloff, 2009; Gupta and Ji, 2009; Liao and Grishman, 2010b,a; Huang and Riloff, 2012b; Li et al., 2013) focus on designing hand-crafted features and heuristic rules to extract event arguments, which suffer from the problem of both implementation complexity and low recall. As the rapid develop-

ment of neural networks, various neural methods have been proposed, such as utilizing convolutional models (Chen et al., 2015), utilizing recurrent models (Nguyen et al., 2016; Sha et al., 2018), and fine-tuning pre-trained language model BERT (Wang et al., 2019b). As compared with the early feature-based and rule-based methods, neural methods automatically represent sentence semantics with low-dimensional vectors, and independently determine argument roles with the vectors, leading to getting rid of designing sophisticated features and rules. Recently, some works adopt some advanced techniques to further improve EAE models in different scenarios, including zero-shot learning (Huang et al., 2018), multi-modal integration (Zhang et al., 2017), cross-lingual (Subburathinam et al., 2019), end-to-end (Wadden et al., 2019), and weak supervision (Chen et al., 2017; Zeng et al., 2018).

The current methods for EAE have achieved some promising results. However, they focus on independently handling each argument entity to predict its role. Because of ignoring to capture rich correlated knowledge among event arguments, the above-mentioned methods are easy to trap in a local optimum and make some inexplicable mistakes. Inspired by some methods in named entity recognition (Huang et al., 2015) and relation extraction (Miwa and Bansal, 2016), some recent proactive works view EAE as a sequence labeling problem. Following the methods for sequence labeling problem (Ma and Hovy, 2016), these sequential EAE models (Yang and Mitchell, 2016; Zeng et al.,

2018) adopt conditional random field (CRF) with the Viterbi algorithm (Rabiner, 1989), and unintentionally consider the correlation of event arguments. Limited by the Markov property, the linear-chain CRF sequentially considers the correlation between two adjacent event arguments, which cannot adequately handle the complex situation in EAE that each argument and any other arguments may be correlated. To this end and inspired by some proactive works (Finkel et al., 2005; Sun et al., 2014), we adapt Gibbs sampling (Geman and Geman, 1987) for EAE to perform approximate inference from the joint distribution. Moreover, we incorporate simulated annealing (Kirkpatrick et al., 1983) to accelerate the sampling process, leading to an effective and efficient method.

### 3 Methodology

#### 3.1 Framework

For convenience, we denote  $X = \{x_1, \dots, x_n\}$  and  $X_{-i} = \{x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n\}$ . Figure 2 shows the overall framework of our Neural Gibbs Sampling (NGS) method, consisting of the following modules:

**The neural models**, including a prior neural model to model the prior distribution  $P_p(x_i|o)$ , and a conditional neural model to model the conditional distribution  $P_c(x_i|X_{-i}, o)$ . The prior neural model is similar with existing EAE methods, which takes the event mention text as input and outputs the labels of event argument candidates. The labels will serve as the prior state for the Gibbs sampling module. The conditional neural model takes the text and the results of the last step as input and outputs the probability distribution over labels for each event argument candidate.

**The Gibbs sampling module** to sample variable assignments  $X$  with  $P_p(x_i|o)$  and  $P_c(x_i|X_{-i}, o)$ , which gradually match the implicit posterior joint distribution.

**The simulated annealing method** to efficiently find the optimal state in the Markov chain of Gibbs sampling. It uses a “temperature” parameter to control the sharpness of the transition distribution. With the “temperature” decreasing, the algorithm will more and more tend to choose the max-likelihood state as the next state.

#### 3.2 Neural Models

**The Prior Neural Model** is to model the prior distribution  $P_p(x_i|o)$ . In this paper, we use DM-

CNN (Chen et al., 2015) and DMBERT as the prior neural models. Given a sentence consisting of several words  $\{w_1, \dots, t, \dots, w_i, \dots, w_n\}$ , where  $t$  and  $w_i$  denote the trigger word and the candidate argument entity respectively.

**DMCNN** transfers each word in the word sequence into an input embedding  $e_i$ , which consists of word embedding, event type embedding, and position embedding. Then, DMCNN feeds the input embeddings into a convolutional encoding layer to automatically learn the features and a dynamic multi-pooling layer to aggregate the features into a unified sentence observation embedding to predict an argument role  $x_i$  for  $w_i$ .

**DMBERT** is a variation of BERT (Devlin et al., 2019) proposed by Wang et al. (2019b). It adopts a pre-trained BERT to represent the word sequence as feature vectors and also uses a dynamic multi-pooling mechanism like DMCNN to aggregate the features into an instance embedding for prediction. It inserts special tokens around the event argument candidates to indicate their positions.

We sample an argument role following  $P_p(x_i|o)$  for each argument candidate and finally predict an initial argument role state  $X^{(0)} = \{x_1^{(0)}, \dots, x_n^{(0)}\}$  as the start point of Gibbs sampling. Note that, our NGS method does not have any special requirements for the prior neural model, any other neural networks can also be used.

**Conditional Neural Model** is to model the conditional distribution  $P_c(x_i|X_{-i}, o)$  for the state transition in Gibbs sampling. Considering that it requires to integrate the argument role information of  $X_{-i}$  to compute  $P_c(x_i|X_{-i}, o)$ , we set an argument role embedding  $a_i$  for each word  $w_i$  to represent whether it is an event argument and which role it is of. Then, we modify the input layer of DMCNN and DMBERT to feed the argument role embeddings in. More specifically, DMCNN concatenates the original input embedding  $e_i$  with the argument role embedding  $a_i$  as new inputs. DMBERT utilizes the pre-trained parameters and adds  $a_i$  into the input embedding.

#### 3.3 Gibbs Sampling Module

The Gibbs sampling module aims at sampling from the implicit joint distribution  $P(X|o)$ . As Algorithm 1 shows, we use the prior neural model to initialize an initial state  $X^{(0)}$ . In step  $t$ , for each random variable  $x_i$ , we input the other random variables’ states  $X_{-i}^{(t-1)}$  into the conditional neu-

---

**Algorithm 1** Neural Gibbs sampling

---

**Input:** Initial state  $X^{(0)} = \{x_1^{(0)}, \dots, x_n^{(0)}\}$  predicted by the prior neural network  
**Result:**  $N$  samples matching the joint distribution  $P(X|o)$   
Train the conditional neural model to fit  $P_c(x_i|X_{-i}, o)$   
**for**  $t \leftarrow 1$  **to**  $N$  **do**  
    // iteratively change the state  
    **for**  $i \leftarrow 1$  **to**  $n$  **do**  
         $x_i^{(t)} \leftarrow \text{sample} \left( P_c(x_i^{(t)}|X_{-i}^{(t-1)}, o) \right)$   
    **end**  
     $X^{(t)} \leftarrow \{x_1^{(t)}, \dots, x_n^{(t)}\}$   
**end**  
Return  $X^{(1)}, \dots, X^{(N)}$

---

ral model to get the distribution  $P_c(x_i^{(t)}|X_{-i}^{(t-1)}, o)$ . Then we sample  $x_i^{(t)}$  from the distribution, and finally get the new state  $X^{(t)}$ . We can approximately sample  $N$  samples  $X^{(1)}, \dots, X^{(N)}$  with the Gibbs sampling module. Our Appendix gives the proof that the samples will accurately follow the joint distribution after enough steps.

Geman and Geman (1987) have shown that the samples from the beginning of the Markov chain (the burn-in period) may not accurately follow the desired distribution, hence we choose the most frequent state from  $X^{(\frac{N}{2})}, \dots, X^{(N)}$  as the result.

### 3.4 Simulated Annealing Method

The Gibbs sampling module is to accurately estimate the shape of  $P(X|o)$ , which will take many steps to reach the convergence. As what we want for EAE is only the max-likelihood state, we adopt a simulated annealing method to efficiently find the optimal state following Geman and Geman (1987).

As shown in Algorithm 2, in step  $t$ , the simulated annealing method randomly sample an  $i$  from the distribution  $\frac{\max(P_c(x_i^{(t)}|X_{-i}^{(t-1)}, o))^{1/c}}{\sum_{j=1}^n \max(P_c(x_j^{(t)}|X_{-j}^{(t-1)}, o))^{1/c}}$ . The probability of  $i$  being chosen has positive correlation with the probability of the max-likelihood state in the conditional distribution of  $x_i$ . Then we only need to update  $x_i$  with its max-likelihood state in conditional distribution  $P_c(x_i^{(t)}|X_{-i}^{(t-1)})$  modeled by the conditional neural model to get the next state  $X^{(t)}$ , which is more efficient than the original Gibbs sampling method. The simulated annealing method adopts a time-varying parameter  $c$

---

**Algorithm 2** NGS + simulated annealing

---

**Input:** Initial state  $X^{(0)} = \{x_1^{(0)}, \dots, x_n^{(0)}\}$  predicted by the prior neural network  
**Result:** The max-likelihood state  $X^{(N)}$   
Train the conditional neural model to fit  $P_c(x_i|X_{-i}, o)$   
 $c = 1$   
**for**  $t \leftarrow 1$  **to**  $N$  **do**  
    // randomly choose  $i$  to transit  
     $i \leftarrow \text{sample} \left( \frac{\max(P_c(x_i^{(t)}|X_{-i}^{(t-1)}, o))^{1/c}}{\sum_{j=1}^n \max(P_c(x_j^{(t)}|X_{-j}^{(t-1)}, o))^{1/c}} \right)$   
     $x_i^{(t)} \leftarrow \arg \max \left( P_c(x_i^{(t)}|X_{-i}^{(t-1)}, o) \right)$   
     $X^{(t)} \leftarrow X_{-i}^{(t-1)} \cup \{x_i^{(t)}\}$   
    decrease  $c$   
**end**  
Return  $X^{(N)}$

---

to control the sharpness of the distribution. With  $c$  gradually decreasing, the algorithm more and more tends to transit in the max-likelihood way and will quickly reach the max-likelihood state. When  $c$  is large, it performs like the original Gibbs sampling, so that can avoid falling into suboptimal results.

## 4 Experiments

### 4.1 Datasets and Evaluation Metrics

We evaluate the proposed models on two real-world datasets: the most widely-used ACE 2005 (Walker et al., 2006) and the newly-developed TAC KBP 2016 (Ellis et al., 2015). They are both often used as the benchmark in the previous works.

**ACE 2005**<sup>1</sup> is the most widely-used dataset in EE, consisting of 599 documents, 8 event types, 33 event subtypes, and 35 argument roles. We evaluate our models by the performance of argument classification. When testing models, an argument is correctly classified only if its event subtype, offsets and argument role match the annotation results. For fair comparison with the previous works (Liao and Grishman, 2010b; Chen et al., 2015), we follow them to use the same test set containing 40 newswire documents, the similar development set with 30 randomly selected documents and training set with the remaining 529 documents.

**TAC KBP 2016**<sup>2</sup> indicates the data of the TAC KBP 2016 Event Argument Extraction track, which is the latest benchmark dataset in EE. Different

<sup>1</sup><https://catalog.ldc.upenn.edu/LDC2006T06>

<sup>2</sup><https://tac.nist.gov/2016/KBP/>

from ACE 2005, this competition only annotates difficult test data but no training data. Accordingly, they encourage participants to construct training data from any other sources by themselves. Considering the argument roles of TAC KBP 2016 are almost the same with ACE 2005 expect TAC KBP 2016 merges all the time-related roles in ACE 2005. We use the ACE 2005 dataset as our training data, which is also provided to the participants of the competition. Hence we can have a fair comparison with the baselines.

For fair comparison with the baselines, we use the same evaluation metrics with previous works: (1) **Precision (P)**, which is defined as the number of correct argument predictions divided by the number of all argument predictions returned by the model. (2) **Recall (R)**, which defined as the number of correct argument predictions divided by the number of all correct golden results in the test set. (3) **F1 score (F1)**, which is defined as the harmonic mean of the precision and recall. F1 score is the most important metric to evaluate EAE performance.

## 4.2 Baselines

To directly show the improvement of our method from the comparisons, we reproduce **DMCNN** and **DMBERT** as baselines on both of the two datasets. In addition, we also select some state-of-the-art baselines on the two datasets respectively.

On **ACE 2005**, we compare our models with various state-of-the-art baselines, including: (1) Feature-based methods. **Li’s joint** (Li et al., 2013) adopts structure prediction to extract events, which is the best traditional feature-based method. **RBPB** (Sha et al., 2016) adopts a regularization-based method to balance the effect of features and patterns, and also consider the relationship between argument candidates. (2) Vanilla neural network methods. **JRNN** (Nguyen et al., 2016) jointly conducts event detection and event argument extraction with bidirectional recurrent neural networks. (3) Advanced neural network method with external information. The **dbRNN** (Sha et al., 2018) utilizes a recurrent neural network with dependency bridges to carry syntactically related information between words, which considers not only sequence structures but also tree structures of the sentences. The **HMEAE** (Wang et al., 2019b) leverages the latent concept hierarchy among argument roles with neural module networks, which considers the label

Learning Rate	$10^{-3}$
Batch Size	60
Dropout Probability	0.5
Hidden Layer Dimension	300
Kernel Size	3
Word Embedding Dimension	100
Position Embedding Dimension	5
Event Type Embedding Dimension	5
Argument Role Embedding Dimension	5

Table 1: Hyperparameter settings for CNN models.

Learning Rate	$6 \times 10^{-5}$
Batch Size	50
Warmup Rate for the Prior Neural Model	0.1
Warmup Rate for the Conditional Nueral Model	0.05
Argument Role Embedding Dimension	768

Table 2: Hyperparameter settings for BERT models.

dependency but still classify each event argument independently.

On **TAC KBP 2016**, we compare our models with the top systems of the competition, including: **DISCERN-R** (Dubbin et al., 2016), **CMU CS Event1** (Hsi et al., 2016), **Washington1** and **Washington4** (Ferguson et al., 2016).

## 4.3 Hyperparameter Settings

Our methods with DMCNN and DMBERT as the prior and conditional neural networks are named as **NGS (CNN)** and **NGS (BERT)** respectively. They both transit for 200 steps and the  $c$  linearly decrease from 1 to 0. As our work focuses on extracting event arguments and their roles and our methods do not involve the event detection stage (to identify the trigger and determine the event type), we conduct EAE based on the event detection models in (Chen et al., 2015) and (Wang et al., 2019a) for the CNN and BERT models respectively.

For **NGS (CNN)**, the hyperparameters of the prior and conditional neural networks are set as the same as in the original **DMCNN** (Chen et al., 2015). We also use the pre-trained word embeddings learned by Skip-Gram (Mikolov et al., 2013) as the initial word embeddings. The detailed hyperparameters are shown in Table 1.

For **NGS (BERT)**, the two BERT models for the prior and conditional probability distributions are both based on the BERT<sub>BASE</sub> model in Devlin et al. (2019). We apply the pre-trained model <sup>3</sup> to initialize the parameters. To utilize the event type information in our model, we append a special token into each input sequence for BERT to indicate

<sup>3</sup>[github.com/google-research/bert](https://github.com/google-research/bert)

Method	Trigger Classification			Argument Role Classification		
	P	R	F1	P	R	F1
Li's Joint	73.7	62.3	67.5	64.7	44.4	52.7
DMCNN	75.6	63.6	69.1	62.2	46.9	53.5
RBPB	70.3	67.5	68.9	54.1	53.5	53.8
JRNN	66.0	73.0	69.3	54.2	56.7	55.4
HMEAE (CNN)	75.6	63.6	69.1	57.3	54.2	55.7
DMBERT	77.6	71.8	74.6	58.8	55.8	57.2
dbRNN	74.1	69.8	71.9	66.2	52.8	58.7
HMEAE (BERT)	77.6	71.8	74.6	62.2	56.6	59.3
NGS (CNN)	75.6	63.6	69.1	61.3	51.3	55.9
NGS (BERT)	77.6	71.8	74.6	59.9	59.1	59.5

Table 3: The overall EAE results (%) of various baselines and NGS on ACE 2005. EAE performances are influenced by the trigger quality, hence we also provide the trigger classification (event detection) results. Note that as our work does not involve the event detection stage, the NGS (CNN) and NGS (BERT) use the triggers predicted by DMCNN and DMBERT respectively.

Method	Argument Role Classification		
	P	R	F1
DISCERN-R (Dubbin et al., 2016)	7.9	7.4	7.7
Washington4 (Ferguson et al., 2016)	32.1	5.0	8.7
CMU CS Event1 (Hsi et al., 2016)	31.2	4.9	8.4
Washington1 (Ferguson et al., 2016)	26.5	6.8	10.8
DMCNN (Chen et al., 2015)	17.9	16.0	16.9
HMEAE (CNN) (Wang et al., 2019b)	15.3	22.5	18.2
DMBERT (Wang et al., 2019b)	22.6	24.7	23.6
HMEAE (BERT) (Wang et al., 2019b)	24.8	25.4	25.1
NGS (CNN)	21.5	16.2	18.5
NGS (BERT)	25.5	25.1	25.3

Table 4: The overall EAE results (%) of various baseline methods and our NGS on TAC KBP 2016 Event Argument Task. All the models use golden triggers.

the event type. Additional hyperparameters used in our experiments are shown in Table 2.

#### 4.4 Overall Evaluation Results

The overall results of various baseline methods and NGS on ACE 2005 are shown in Table 3. And the results on TAC KBP 2016 are shown in Table 4. From the results, we observe that:

(1) NGS (CNN) and NGS (BERT) achieve significant improvements as compared with DMCNN and DMBERT respectively. Meanwhile, our models still outperform other baseline methods, which are either the typical EAE models or the recent state-of-the-art models. It indicates that our Gibbs sampling with simulated annealing works well to improve EAE with the help of adequately model-

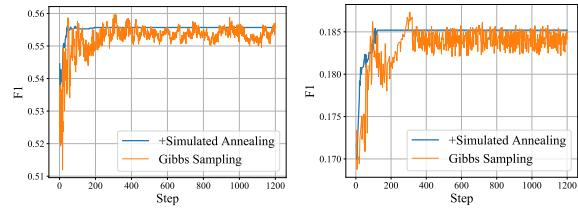


Figure 3: F1-step curves of NGS (CNN) with the simulated annealing method and the original Gibbs sampling on ACE 2005 (left) and TAC KBP 2016 (right).

ing the correlation between event arguments. This demonstrates that our method is effective.

(2) As NGS enhances both CNN models and BERT models on different datasets, it shows that our Gibbs sampling with simulated annealing is independent of EAE models. In other words, our method can be easily adapted for other EAE models to enhance their extraction performances.

(3) From the experimental results on both ACE 2005 and TAC KBP 2016, we can find that the recall scores and F1 scores of our models are much better than the baseline models. The precision scores of our models do not achieve such obvious improvements. This is consistent with what we mention in the previous sections.

We argue that the baseline models focusing on independently handling each event argument candidates may sever the constraints among argument roles, and may trap in a local optimum or over-fit the training set. The models without considering argument correlations may predict various argument roles with high confidence, even make some inexplicable mistakes. Hence the precision scores of these models may increase, but their recall scores and F1 scores may decrease.

Our models adopt Gibbs sampling for EAE to perform approximate inference from the joint distribution, and make the most of the correlation and constraints among argument roles. Accordingly, our models can avoid these issues and achieve the state-of-the-art results.

#### 4.5 Ablation Study

In order to verify the effectiveness of our method, especially for the simulated annealing method and the prior neural network, we conduct ablation studies on ACE 2005 and TAC KBP 2016.

#### Effectiveness of the Simulated Annealing

To demonstrate the effectiveness of the simulated annealing method, we show the F1-step curves of

---

Type: Justice Subtype: Appeal

---

Text: Malaysia’s second highest court on Friday rejected an appeal by ... Anwar Ibrahim against his conviction and nine-year prison sentence for sodomy.

---

Event Argument Candidate	Malaysia	court	Friday	Anwar Ibrahim	sodomy
DMCNN	Place✓	Adjudicator✓	Time-Within✓	Plaintiff✓	N/A×
NGS (CNN)	Place✓	Adjudicator✓	Time-Within✓	Plaintiff✓	Crime✓

---

Table 5: Top: An example sentence highlighting the event argument candidates, which is sampled from ACE 2005. Bottom: EAE results of DMCNN and NGS (CNN). NGS (CNN) correctly classifies “sodomy” into Crime with the help of correlations among event arguments.

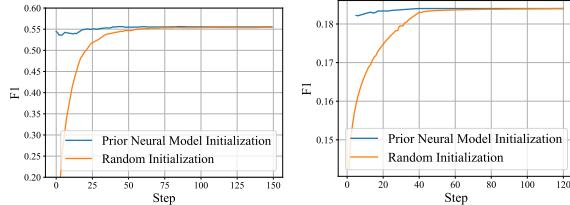


Figure 4: F1-step curves of NGS (CNN) with prior neural network initialization and random initialization on ACE 2005 (left) and TAC KBP 2016 (right).

Gibbs sampling with and without the simulated annealing in Figure 3. We can observe that:

(1) The simulated annealing method can significantly improve the convergence speed and the stability. Our methods just require quarter to half of the steps to reach the convergence.

(2) The simulated annealing method does not weaken the performance of our models. Although the methods with the simulated annealing are much more efficient than those without the simulated annealing, their results are comparable.

### Effectiveness of the Prior Neural Network

As the mathematical proof in the Appendix shows, a prior distribution is not necessary for Gibbs sampling. To demonstrate the effectiveness of the prior neural model, we show the F1-step curves of the prior neural model initialization and a random initialization for our NGS method (with simulated annealing) in Figure 4. As it shows in figures, our NGS models with the prior neural network initialization take much fewer steps to reach the convergence than those models with random initialization, which is important and meaningful for the application. Combining the prior neural network initialization and the simulated annealing for our NGS will lead to a more efficient model.

#arguments	1-2	3-4	>5
DMCNN	55.3	54.1	61.8
NGS (CNN)	56.7 (+1.4)	57.9 (+3.8)	69.5 (+7.7)

Table 6: F1 scores (%) of DMCNN and NGS (CNN) on different parts of ACE 2005 dev set with different event argument numbers per sentence.

### 4.6 Analysis on Modeling Event Argument Correlations

To analyze whether NGS can successfully capture the event argument correlations and further improve EAE performance, we conduct a case study in Table 5 and a quantitative analysis in Table 6.

The sentence in Table 5 is a real sentence containing an Appeal event, which is sampled from the test set of ACE 2005. From the EAE results, we can see that the vanilla DMCNN correctly classifies most of the event argument candidates. But because “sodomy” is a rare word, it misclassified “sodomy” into “N/A” (not an event argument). With the help of our NGS method’s ability to model the joint distribution among event arguments, NGS (CNN) can infer that “sodomy” is a crime from the event argument correlations as it has known there are some crime-related arguments (adjudicator and plaintiff) in the sentence.

On the other side, we show the comparisons between the basic model DMCNN and NGS (CNN) on data with different numbers of event arguments in Table 6. With the increase of event argument number, our improvements significantly rise, which demonstrates our improvements come from modeling the correlations among event arguments. Note that the F1 scores are higher than the overall F1 scores, which is due to we filter out the negative instances without event arguments.

## 5 Conclusion and Future Work

In this paper, we propose a novel Neural Gibbs Sampling (NGS) method to adequately model the correlation between event arguments and argument roles, which combines the advantages of the Gibbs sampling method to model the joint distribution among random variables and the neural network models to automatically learn the effective representations. Considering the shortcoming of high complexity of Gibbs sampling algorithm, we further apply simulated annealing to accelerate the whole estimation process, which lead our method to being both effective and efficient.

The experimental results on two widely-used real-world datasets show that NGS can achieve comparable results to existing state-of-the-art EAE methods. The empirical analyses and ablation studies further verify the effectiveness and efficiency of our method. In the future: (1) We will try to extend NGS to other tasks and scenarios to evaluate its general effectiveness of modeling the latent correlations. (2) We will also explore more effective and simple methods to consider the correlations.

## Acknowledgement

We thank Hedong (Ben) Hou for his help in the mathematical proof. This work is supported by the Key-Area Research and Development Program of Guangdong Province (2019B010153002), NSFC Key Projects (U1736204, 61533018), a grant from Institute for Guo Qiang, Tsinghua University (2019GQB0003) and THUNUS NExT Co-Lab. This work is also supported by the Pattern Recognition Center, WeChat AI, Tencent Inc. Xiaozhi Wang is supported by Tsinghua University Initiative Scientific Research Program.

## References

- David Ahn. 2006. *The stages of event extraction*. In *ARTE*.
- Jun Araki and Teruko Mitamura. 2015. *Joint event trigger identification and event coreference resolution with structured perceptron*. In *EMNLP*.
- P Basile, A Caputo, G Semeraro, and L Siciliani. 2014. *Extending an information retrieval system through time event extraction*. In *DART*.
- Yubo Chen, Shulin Liu, Xiang Zhang, Kang Liu, and Jun Zhao. 2017. *Automatically labeled data generation for large scale event extraction*. In *ACL*.
- Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. *Event extraction via dynamic multi-pooling convolutional neural networks*. In *ACL-IJCNLP*.
- Pengxiang Cheng and Katrin Erk. 2018. *Implicit argument prediction with event knowledge*. In *ACL*.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. *BERT: Pre-training of deep bidirectional transformers for language understanding*. In *NAACL-HLT*.
- Shaoyang Duan, Ruifang He, and Wenli Zhao. 2017. *Exploiting document level information to improve event detection via recurrent neural networks*. In *IJCNLP*.
- Greg Dubbin, Archna Bhatia, Bonnie Dorr, Adam Dalton, Kristy Hollingshead, Suriya Kandaswamy, Ian Perera, and Jena D Hwang. 2016. *Improving discern with deep learning*. In *TAC*.
- Joe Ellis, Jeremy Getman, Dana Fore, Neil Kuster, Zhiyi Song, Ann Bies, and Stephanie M Strassel. 2015. *Overview of linguistic resources for the tac kbp 2016 evaluations: Methodologies and results*. In *TAC*.
- James Ferguson, Colin Lockard, Natalie Hawkins, Stephen Soderland, Hannaneh Hajishirzi, and Daniel S Weld. 2016. *University of washington tac-kbp 2016 system description*. In *TAC*.
- Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. *Incorporating non-local information into information extraction systems by Gibbs sampling*. In *ACL*.
- Stuart Geman and Donald Geman. 1987. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. In *Readings in computer vision*.
- Reza Ghaeini, Xiaoli Fern, Liang Huang, and Prasad Tadepalli. 2016. *Event nugget detection with forward-backward recurrent neural networks*. In *ACL*.
- Prashant Gupta and Heng Ji. 2009. *Predicting unknown time arguments based on cross-event propagation*. In *ACL-IJCNLP*.
- Yu Hong, Jianfeng Zhang, Bin Ma, Jianmin Yao, Guodong Zhou, and Qiaoming Zhu. 2011. *Using cross-entity inference to improve event extraction*. In *ACL-HLT*.
- Yu Hong, Wenxuan Zhou, Guodong Zhou, Qiaoming Zhu, et al. 2018. *Self-regulation: Employing a generative adversarial network to improve event detection*. In *ACL*.
- Andrew Hsi, Jaime G Carbonell, and Yiming Yang. 2016. *Cmu cs event tac-kbp2016 event argument extraction system*. In *TAC*.

- Lifu Huang, Heng Ji, Kyunghyun Cho, Ido Dagan, Sebastian Riedel, and Clare Voss. 2018. Zero-shot transfer learning for event extraction. In *ACL*.
- Ruihong Huang and Ellen Riloff. 2012a. Bootstrapped training of event extraction classifiers. In *Proceedings of EACL*.
- Ruihong Huang and Ellen Riloff. 2012b. Modeling textual cohesion for event extraction. In *AAAI*.
- Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *ArXiv*.
- Heng Ji and Ralph Grishman. 2008. Refining event extraction through cross-document inference. In *ACL*.
- Scott Kirkpatrick, C Daniel Gelatt, and Mario P Vecchi. 1983. Optimization by simulated annealing. *Science*.
- Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *ACL*.
- Shasha Liao and Ralph Grishman. 2010a. Filtered ranking for bootstrapping in event extraction. In *COLING*.
- Shasha Liao and Ralph Grishman. 2010b. Using document level cross-event inference to improve event extraction. In *ACL*.
- Hongyu Lin, Yaojie Lu, Xianpei Han, and Le Sun. 2018. Nugget proposal networks for chinese event detection. In *ACL*.
- Jian Liu, Yubo Chen, Kang Liu, and Jun Zhao. 2018a. Event detection via gated multilingual attention mechanism. In *AAAI*.
- Shaobo Liu, Rui Cheng, Xiaoming Yu, and Xueqi Cheng. 2018b. Exploiting contextual information via dynamic memory network for event detection. In *EMNLP*.
- Shulin Liu, Yubo Chen, Shizhu He, Kang Liu, and Jun Zhao. 2016a. Leveraging framenet to improve automatic event detection. In *ACL*.
- Shulin Liu, Yubo Chen, Kang Liu, and Jun Zhao. 2017. Exploiting argument information to improve event detection via supervised attention mechanisms. In *ACL*.
- Shulin Liu, Kang Liu, Shizhu He, and Jun Zhao. 2016b. A probabilistic soft logic based approach to exploiting latent and global information in event classification. In *AAAI*.
- Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *ACL*.
- David McClosky, Mihai Surdeanu, and Christopher D Manning. 2011. Event extraction as dependency parsing. In *ACL*.
- Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *ICLR*.
- Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using lstms on sequences and tree structures. In *ACL*.
- Thien Nguyen and Ralph Grishman. 2018. Graph convolutional networks with argument-aware pooling for event detection. In *AAAI*.
- Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. Joint event extraction via recurrent neural networks. In *NAACL-HLT*.
- Thien Huu Nguyen and Ralph Grishman. 2015. Event detection and domain adaptation with convolutional neural networks. In *ACL-IJCNLP*.
- E. Nummerlin. 1984. *General Irreducible Markov Chains and Non-negative Operators*.
- Siddharth Patwardhan and Ellen Riloff. 2009. A unified model of phrasal and sentential evidence for information extraction. In *EMNLP*.
- Lawrence R Rabiner. 1989. A tutorial on hidden markov models and selected applications in speech recognition. *IEEE*.
- Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *ECML-PKDD*.
- Lei Sha, Jing Liu, Chin-Yew Lin, Sujian Li, Baobao Chang, and Zhifang Sui. 2016. RBPB: Regularization-based pattern balancing method for event extraction. In *ACL*.
- Lei Sha, Feng Qian, Baobao Chang, and Zhifang Sui. 2018. Jointly extracting event triggers and arguments by dependency-bridge rnn and tensor-based argument interaction. In *AAAI*.
- Ananya Subburathnam, Di Lu, Heng Ji, Jonathan May, Shih-Fu Chang, Avirup Sil, and Clare Voss. 2019. Cross-lingual structure transfer for relation and event extraction. In *Proceedings of EMNLP-IJCNLP*, pages 313–325.
- Liang Sun, Jason Mielens, and Jason Baldridge. 2014. Parsing low-resource languages using Gibbs sampling for PCFGs with latent annotations. In *EMNLP*.
- L. Tierney. 1991. Ace 2005 multilingual training corpus. *Tech. Rept., School of Statist., Univ. of Minnesota*.
- David Wadden, Ulme Wennberg, Yi Luan, and Hanneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. In *Proceedings of EMNLP-IJCNLP*, pages 5784–5789.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. *Ace 2005 multilingual training corpus*. *Linguistic Data Consortium, Philadelphia*.

Xiaozhi Wang, Xu Han, Zhiyuan Liu, Maosong Sun, and Peng Li. 2019a. *Adversarial training for weakly supervised event detection*. In *NAACL-HLT*.

Xiaozhi Wang, Ziqi Wang, Xu Han, Zhiyuan Liu, Juanzi Li, Peng Li, Maosong Sun, Jie Zhou, and Xiang Ren. 2019b. *HMEAE: Hierarchical modular event argument extraction*. In *EMNLP-IJCNLP*.

Bishan Yang and Tom Mitchell. 2016. *Joint extraction of events and entities within a document context*. In *NAACL-HLT*.

Hui Yang, Tat-Seng Chua, Shuguang Wang, and Chun-Keat Koh. 2003. *Structured use of external knowledge for event-based open domain question answering*. In *SIGIR*.

Ying Zeng, Yansong Feng, Rong Ma, and Zheng Wang. 2018. *Scale up event extraction learning via automatic training data generation*. In *AAAI*.

Tongtao Zhang, Spencer Whitehead, Hanwang Zhang, Hongzhi Li, Joseph Ellis, Lifu Huang, Wei Liu, Heng Ji, and Shih-Fu Chang. 2017. *Improving event extraction via multimodal integration*. In *MM*.

Yue Zhao, Xiaolong Jin, Yuanzhuo Wang, and Xueqi Cheng. 2018. *Document embedding enhanced event detection with hierarchical and supervised attention*. In *ACL*.

## A Proof of the Convergence of Gibbs Sampling

In this section, we will prove the convergence of Gibbs sampling, by which we implement sampling from the implicit joint distribution in this paper.

Suppose that  $X = (X_0, \dots, X_n, \dots)$ ,  $X_i \in E \subseteq \mathbf{R}^n$  is a Markov chain (abbr. MC). For a  $\nu$ -measurable set  $A$ , the transition kernel of  $A$ ,  $K : E \times E \rightarrow \mathbf{R}^n$  is defined via the following equation,

$$K(X_i, A) = \mathbf{P}\{X_{i+1} \in A | X_0, \dots, X_i\} \quad (1)$$

Assume that  $X$  satisfies that for any  $\sigma$ -finite Borel measure  $\nu$  on  $\mathbf{R}^n$ , for any  $\nu$ -measurable set  $A$ , we have that,

$$\begin{aligned} \mathbf{P}(X_i \in A | X_{i-1} = x) &= \\ \int_A K(x, y) d\nu(y) + \chi_A(x)r(x) &\end{aligned} \quad (2)$$

where

$$r(x) := 1 - \int_E K(x, y) d\nu(y)$$

A fundamental property of  $K$  is sub-stochastic. Assume that  $K$  is non-degenerate, hence  $r(x) < 1$  for all  $x \in E$ . Then, following the convention, we can define the iterative form as,

$$\begin{aligned} K^{(t)}(x, y) &= \int_{\mathbf{R}^n} K^{(t-1)}(x, z) K(z, y) d\nu(z) \\ &+ K^{(t-1)}(x, y) r(y) + [1 - r(x)]^{t-1} K(x, y) \end{aligned} \quad (3)$$

Define the invariant distribution as  $\pi(X)$  for this MC and  $D = \{x \in E; \pi(x) > 0\}$ . We know that  $\pi(X)$  must satisfy that, for any  $\nu$ -measurable set  $A$ ,

$$\pi(A) = \int P(X_1 \in A | X_0 = x) \pi(x) d\nu(x) \quad (4)$$

For  $\nu$ -measurable  $A$ ,  $K$  is called  **$\pi$ -irreducible** when for all  $x \in D$ ,  $\pi(A) > 0$ , and is called **aperiodic** when there exists no partition  $E = (E_1, \dots, E_{k-1})$  such that  $\mathbf{P}(X_{i+1} \in A_{j+1} | X_i \in A_j) = 1$  for all  $j = 1, \dots, k-1$  ( $\text{mod } k$ ). Due to the work of Nummerlin (1984) and Tierney (1991), we have the following theorem: If  $K$  is  $\pi$ -irreducible and aperiodic then, for all  $x \in D$ .

1.  $|K_x^{(t)} - \pi| \rightarrow 0$  as  $t \rightarrow \infty$ ;
2. for real-valued,  $\pi$ -integrable function  $f$ ,

$$\begin{aligned} t^{-1} \{f(X_1) + \dots + f(X_t)\} \\ \rightarrow \int_E f(x) \pi(x) d\nu(x) \text{ a.s. as } t \rightarrow \infty \end{aligned}$$

where following the conventional transformation between multi-variable functions and parameter families,  $K_x^{(t)}$  is defined as  $K_x^{(t)}(y) := K^{(t)}(x, y)$ . Indeed, with respect to  $\nu$ , it is the density of  $X_t$  provided that  $X_0 = x$ , excluding the realizations  $X_j = x, j = 1, \dots, t$ .

Let  $\mathbf{P}(\mathbf{X}) = \mathbf{P}(X_1, \dots, X_n)$  denote the target density in our case. What we shall prove is that this  $\mathbf{P}(\mathbf{X})$  is the invariant distribution of the MC constructed by Gibbs sampling. Provided with the theorem above, the remaining key issue is to prove that the transition kernel  $K$  satisfies  $\pi$ -irreducibility and aperiodicity.

Equipped with the product measure, for the blocking  $x = (x_1, \dots, x_n)$ , it is required that the conditionals of Gibbs sampler construction,

$$\pi(x_i|x_{-i}) = \frac{\pi(x)}{\int \pi(x) d\nu_i(x_i)}$$

are well-defined over the appropriate regions, where  $\mathbf{X}_{-i}$  shares the same definition as Sec.(2). With  $D = \{x \in E; \pi(x) > 0\}$ , we seek to construct the kernel as  $K : D \times D \rightarrow \mathbf{R}^n$  via

$$K(x, y) = \begin{cases} \prod_{i=1}^n (\pi(y_i|x_{j,j>i}, y_{j,j< i})) & \text{if } \Upsilon \\ 0 & \text{otherwise} \end{cases}$$

where  $\Upsilon$  denotes the condition that

$$\pi(y_1, \dots, y_i, x_{i+1}, \dots, x_n) d\nu_i(y_i) > 0$$

It is then straightforward to check that, when  $K(x, y)$  is well-defined,  $\pi$  is an invariant distribution of the chain attained by  $K$ .

Observe that since we have a discrete distribution, it is trivial that all the subjects here are well-defined. Also the aperiodicity of  $K$  is ensured by the fact that  $K(x, x) > 0$  for all  $x \in D$ .

# Named Entity Recognition in Multi-level Contexts

Yubo Chen Chuhan Wu Tao Qi Zhigang Yuan Yongfeng Huang

Department of Electronic Engineering & BNRIst, Tsinghua University, Beijing 100084, China

{ybch14, wuchuhan15, taoqi.qt}@gmail.com

{zgyuan, yfhuang}@tsinghua.edu.cn

## Abstract

Named entity recognition is a critical task in the natural language processing field. Most existing methods for this task can only exploit contextual information within a sentence. However, their performance on recognizing entities in limited or ambiguous sentence-level contexts is usually unsatisfactory. Fortunately, other sentences in the same document can provide supplementary document-level contexts to help recognize these entities. In addition, words themselves contain word-level contextual information since they usually have different preferences of entity type and relative position from named entities. In this paper, we propose a unified framework to incorporate multi-level contexts for named entity recognition. We use TagLM as our basic model to capture sentence-level contexts. To incorporate document-level contexts, we propose to capture interactions between sentences via a multi-head self attention network. To mine word-level contexts, we propose an auxiliary task to predict the type of each word to capture its type preference. We jointly train our model in entity recognition and the auxiliary classification task via multi-task learning. The experimental results on several benchmark datasets validate the effectiveness of our method.

## 1 Introduction

Named Entity Recognition (NER) is defined as automatically identifying and classifying named entities into specific categories (e.g., person, location, organization) in text. It is a critical task in Natural Language Processing (NLP) and a prerequisite for many downstream tasks, such as entity linking (Luo et al., 2015), relation extraction (Feldman and Rosenfeld, 2006) and question answering (Lee et al., 2006).

NER is usually modeled as a sentence-level sequence labeling task in previous work. For example, Lample et al. (2016) used long-short term

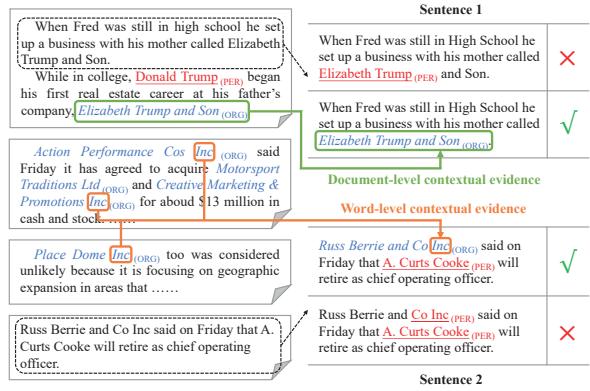


Figure 1: Examples of document- and word-level contextual evidence. Blue italic and red underlined entities are the names of organizations and persons respectively. Green and orange arrows indicate the document- and word-level contextual evidence respectively.

memory (LSTM) (Gers et al., 2000) for capturing contextual word representations and conditional random field (CRF) (Lafferty et al., 2001) for jointly label decoding. In recent years, language models (LMs) were introduced to this task to learn better contextual representations of words (Peters et al., 2017, 2018; Devlin et al., 2019). However, these methods only consider the contexts within a sentence, which is insufficient.

Our work is motivated by the observation that the contextual information beyond sentences can mitigate the negative effects of the ambiguous and limited sentence contexts. The sentences within a document are highly related, and the interactions between them can provide **document-level** contextual information. For example, in Figure 1, sentence 1 is ambiguous because it can be either his mother called *Elizabeth Trump* or a business called *Elizabeth Trump and Son*. But another sentence in this document explicitly mentions *Elizabeth Trump and Son* as a company's name and solves the ambiguity. Besides, words themselves contain prefer-

ences of entity type and relative position from the entities, and the preferences provide **word-level** contextual information. For instance, the sentence 2 in Figure 1 has limited contexts, and the word *said* can easily mislead the classification of the type of *Co Inc*. However, the multiple mentions of *Inc* in other sentences indicate its preference to appear as the last word of organizations. Thus, these preferences of words have the potential to help recognize entity types more correctly.

In this paper, we propose a unified framework for NER to incorporate multi-level contexts. Our framework is based on TagLM (Peters et al., 2017), which captures morphological and sentence-level contextual information with two-layer bidirectional gated recurrent units (BiGRUs) (Chung et al., 2014). We apply the neural attention mechanism (Bahdanau et al., 2014) to the hidden states of TagLM’s bottom BiGRU to learn sentence representations, and contextualize them with a sentence-level BiGRU. To mine document-level contexts, we propose to apply the multi-head self attention mechanism (Vaswani et al., 2017) to the sentence-level BiGRU’s hidden states to capture the relations between sentences. To fuse the document-level context, we combine the output document representations of the self attention module with the corresponding sentence’s bottom hidden states and feed them into TagLM’s top BiGRU. Besides, to mine word-level contextual information, we propose an auxiliary word classifier to predict the probability distributions of word labels because the label distributions describe the type and position preferences of words. The auxiliary word classification task is jointly trained with our NER model via multi-task learning. We concatenate the top BiGRU’s output representations with the output probability vectors of the word classifier to fuse the word-level context and feed them into a CRF for sequence decoding.

The main contributions of this paper are:

- We propose to fuse multi-level contexts for the NER task with a unified framework.
- We propose to exploit the document-level context by capturing the interactions between sentences within a document with the multi-head self attention mechanism.
- We propose to mine the word-level context with an auxiliary word classification task to learn the words’ preferences of entity type and relative position from the entities.
- We conduct experiments on several bench-

mark datasets, and the results validate the effectiveness of our method.

## 2 Related Work

In traditional NER methods, contexts are usually modeled via hand-crafted features. For example, Passos et al. (2014) trained phrase vectors in their lexicon-infused skip-gram model. Lin and Wu (2009) used a linear chain CRF and added phrase cluster features extracted from the web data. However, these methods require heavy feature engineering, which necessities massive domain knowledge. In addition, these methods cannot make full use of contextual information within texts.

In recent years, many neural networks were applied to the NER task. Collobert et al. (2011) first adopted CNNs to learn word representations. Recently, BiLSTM was widely used for long distance context modeling (Chiu and Nichols, 2016; Lample et al., 2016; Ma and Hovy, 2016). Additionally, Chiu and Nichols (2016) employed CNNs to capture morphological word representations; Lample et al. (2016) utilized CRF to model the dependencies between adjacent tags; Ma and Hovy (2016) proposed LSTM-CNNs-CRF model to combine the strengths of these components. Besides, Strubell et al. (2017) proposed iterated-dilated CNNs for higher efficiency than BiLSTM and better capacity with large context than vanilla CNNs. Recent work proved that the context-sensitive representations captured by language models are useful in NER systems. Peters et al. (2017) proposed TagLM model and introduced LM embeddings in this task. Afterwards, ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019) were proposed for better contextual representations. However, these methods focused only on the context within a sentence, so their performance is substantially hurt by the ambiguity and limitation of sentence context.

To combine contexts beyond sentences, several methods were proposed to mine document-level information, such as logical rules (Mikheev et al., 1999), global attention (Xu et al., 2018; Zhang et al., 2018; Hu et al., 2020) and memory mechanisms (Gui et al., 2020). But these methods ignored the sequential characteristics of the sentences within a document, which may be sub-optimal. We observe that contextual associations between sentences in a document have the potential of improving the NER performance. Moreover, the words’ preferences of entity type and relative position from the entities

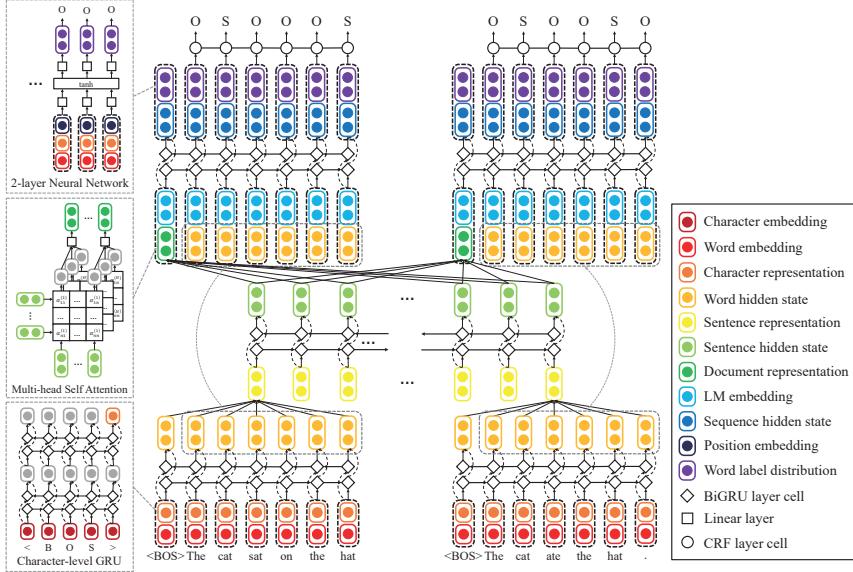


Figure 2: Overview of our multi-level context framework. The character representation is captured with a two-layer BiGRU. The document representation is captured with the multi-head self attention mechanism. The word label distribution is predicted by a two-layer neural network.

contain word-level contextual information, which is ignored by most previous work.

Based on these observations, we propose a unified framework to combine multi-level contexts in this paper. Our framework is based on the TagLM model, which captures sentence-level context with two stacked BiGRUs and models tag dependencies with CRF. To exploit the document-level context, we propose to capture the interactions between sentences within a document with multi-head self attention mechanism (Vaswani et al., 2017). Besides, to mine the word-level context, we propose an auxiliary word classification task to encode the words’ type and position preferences. We train our model in the NER and the auxiliary task via multi-task learning. We conduct experiments on several benchmark datasets, and the results demonstrate the effectiveness of multi-level contexts.

### 3 Our Approach

In this section, we will introduce our approach in detail. The overall framework of our approach is shown in Figure 2. We will first briefly introduce the basic model in our approach, then introduce how to incorporate document- and word-level contexts into our model.

#### 3.1 Baseline NER model

We choose TagLM (Peters et al., 2017) as our basic model. TagLM first captures character-level

information of words because named entities usually have specific morphological patterns. For example, *China* refers to the country in most cases, while *china* mostly refers to porcelains. Therefore, given a sentence of words  $w_1, w_2, \dots, w_n$ , TagLM learns morphological information with a two-layer BiGRU, as shown in Figure 2. It takes the character embeddings (whose dimension denoted as  $d_{ce}$ ) as input, and the last output hidden state is adopted as character representation  $\mathbf{c}_k$ . Then we concatenate  $\mathbf{c}_k$  with a word embedding  $\mathbf{w}_k$  to construct context-independent representation  $\mathbf{x}_k$  for each word:

$$\begin{aligned} \mathbf{c}_k &= \text{BiGRU}(w_k; \theta_c) \in \mathbb{R}^{d_{ch}} \\ \mathbf{w}_k &= E(w_k; \theta_w) \in \mathbb{R}^{d_{we}} \\ \mathbf{x}_k &= [\mathbf{c}_k; \mathbf{w}_k] \in \mathbb{R}^{d_{we}+d_{ch}} \end{aligned} \quad (1)$$

The word embedding  $\mathbf{w}_k$  is obtained by looking up a pre-trained embedding matrix  $\theta_w$ , which is fine-tuned during training (Collobert et al., 2011).

To learn context-sensitive word representations, TagLM applies two layers of BiGRUs on  $[\mathbf{x}_{1:n}]$ . Then the pre-trained LM embeddings are concatenated with the hidden states of the bottom BiGRU. We denote the output of the bottom and the top BiGRU as  $\mathbf{h}_k^{word} \in \mathbb{R}^{d_{sh}}$  and  $\mathbf{h}_k^{seq} \in \mathbb{R}^{d_{sqh}}$ :

$$\begin{aligned} \mathbf{h}_k^{word} &= \text{BiGRU}(\mathbf{x}_k), \\ \mathbf{h}_k^{seq} &= \text{BiGRU}([\mathbf{h}_k^{word}; \mathbf{LM}_k]). \end{aligned} \quad (2)$$

Finally, we feed  $[\mathbf{h}_{1:n}^{seq}]$  into a linear-chain CRF to model the correlations between labels in neighbor-

hoods and jointly decode the best label sequence. The probabilistic model for linear CRF defines a family of conditional probability  $p(\mathbf{y}|\mathbf{z}; \theta)$  over all possible label sequences  $\mathbf{y}$  given  $\mathbf{z}$ :

$$p(\mathbf{y}|\mathbf{z}; \theta) = \frac{\prod_{i=1}^n \psi_i(y_{i-1}, y_i, \mathbf{z})}{\sum_{y' \in \mathcal{Y}(\mathbf{z})} \prod_{i=1}^n \psi_i(y'_{i-1}, y'_i, \mathbf{z})} \quad (3)$$

where  $\psi_i(y', y, \mathbf{z}) = \exp(\mathbf{W}_{y', y}^\top \mathbf{z}_i + \mathbf{b}_{y', y})$  are potential functions, and  $\mathbf{W}_{y', y}, \mathbf{b}_{y', y}$  are parameters of the CRF. Following Lafferty et al. (2001) and Collobert et al. (2011), we utilize the sentence CRF loss for training, which is formulated as the negative log-likelihood:

$$L_{CRF} = - \sum_i \log p(\mathbf{y}|\mathbf{z}; \theta) \quad (4)$$

We compute the likelihood using the forward-backward algorithm at the training phase, and use the Viterbi algorithm to find the most likely label sequence at the test phase.

### 3.2 Document-level Context

Sentences within a document are highly correlated, and these correlations provide contextual information at the document level. For example, in the document “Jason Little is a rugby union player. Little won 75 caps as captain”, the second sentence is ambiguous because it can also mean “Hardly any person won 75 caps as captain”. In this case, the first sentence in this document explicitly mentions *Jason Little* as a player. The interaction between the two sentences helps to solve this ambiguity. Therefore, we capture and fuse the document-level context as follows.

To capture the document-level context, we first obtain the context-independent sentence representations. Since each word in a sentence has different importance (e.g. *a* contributes less information than *player* in “Jason Little is a rugby union player.”), we apply the neural attention mechanism (Bahdanau et al., 2014) to filter the uninformative words and learn better sentence representations. Then we contextualize these representations with a sentence-level BiGRU. Formally,

$$\begin{aligned} \alpha_k &= \text{softmax}(\mathbf{u}_w^\top \cdot \tanh(\mathbf{W}_a \mathbf{h}_k^{word} + \mathbf{b}_a)) \\ \mathbf{s}_i &= \sum_{k=1}^n \alpha_k \mathbf{h}_{ik}^{word} \\ \mathbf{h}_i^{sen} &= \text{BiGRU}(\mathbf{s}_i) \end{aligned} \quad (5)$$

where  $\mathbf{W}_a \in \mathbb{R}^{d_{na} \times d_{wh}}$ ,  $\mathbf{b}_a \in \mathbb{R}^{d_{na}}$ ,  $\mathbf{u}_w \in \mathbb{R}^{d_{na}}$  are the parameters of the neural attention module.

Next, we propose to capture the interactions between sentences with the multi-head self attention mechanism (Vaswani et al., 2017). In most existing attention mechanisms, a sentence’s attention weight is only based on its representation, and the relationships between sentences cannot be modeled. Self attention is an effective way to capture the interactions between sentences. Besides, a sentence may interact with multiple sentences. For example, in the document “LeBron James is a basketball player for the Lakers. In 2016 James won the championship of NBA. In 2018 he signed with the Lakers”, the first sentence interacts with the remaining two sentences simultaneously because they jointly mention *James* and *Lakers* respectively. Thus, we propose to apply the multi-head self attention mechanism to learn better representations of sentences by modeling their relationship with multiple sentences. We first project the sentence hidden states into the  $h$ -th sub-space, and calculate the attention weights in this sub-space:

$$\begin{aligned} [Q_j^{(h)}; K_j^{(h)}; V_j^{(h)}] &= [W_Q^{(h)}; W_K^{(h)}; W_V^{(h)}] \mathbf{h}_j^{sen} \\ z_{ij}^{(h)} &= Q_i^{(h)\top} K_j^{(h)}, \quad \beta_{ij}^{(h)} = \frac{\exp(z_{ij}^{(h)})}{\sum_j \exp(z_{ij}^{(h)})} \end{aligned} \quad (6)$$

Then we calculate the sub-representation  $\mathbf{y}_i^{(h)}$  for the  $i$ -th sentence by weighted summing the  $V_j^{(h)}$ . Finally, these sub-representations are concatenated and projected, resulting in the final representation  $\mathbf{d}_i$  for the  $i$ -th sentence. We denote the number of heads as  $H$  and the sub-space dimension of each head as  $d_{sa}$ , then we have:

$$\begin{aligned} \mathbf{y}_i^{(h)} &= \sum_j \beta_{ij}^{(h)} V_j^{(h)} \\ \mathbf{d}_i &= W_O[\mathbf{y}_i^{(1)}; \dots; \mathbf{y}_i^{(h)}; \dots; \mathbf{y}_i^{(H)}] \end{aligned} \quad (7)$$

where  $W_Q^{(h)}, W_K^{(h)}, W_V^{(h)} \in \mathbb{R}^{d_{sa} \times d_{sh}}$ ,  $W_O \in \mathbb{R}^{d_{sh} \times H d_{sa}}$  are projection matrices.  $\mathbf{d}_i$  combines representations of all sentences within this document, thus is regarded as the document representation for the  $i$ -th sentence.

To fuse the document-level context, we first add a special token  $\langle \text{BOS} \rangle$  (denoted as  $w_{i0}$ ) at the beginning of the sentence  $w_{i1}, \dots, w_{in}$ , and feed the sentence into TagLM’s bottom BiGRU to compute  $[\mathbf{h}_{i0}^{word}, \mathbf{h}_{i1}^{word}, \dots, \mathbf{h}_{in}^{word}]$ . Next we compute the document representation  $\mathbf{d}_i$  and replace  $\mathbf{h}_{i0}^{word}$  with it (requires  $d_{wh} = d_{sh}$ ). Then we feed them into

the top BiGRU. The input of the top BiGRU contains document- and sentence-level contextual representations simultaneously. Thus its output hidden states act as the fusion of the two contexts.

### 3.3 Word-level Context

In natural language, words themselves have different preferences on different entity types and relative positions from the entities. These preferences provide word-level contextual information for the NER task. For example, in the sentence “With only one match before New Year, Real will spend Christmas ahead of others”, the type of the entity *Real* is uncertain because the context of the sentence is inadequate. However, *Real* prefers to appear as the first word of organizations (e.g. *Real Madrid*, *Real Betis* are football clubs). This preference helps to ensure the entity type of *Real*. Thus we learn and incorporate the word-level context as follows.

To learn the word-level context, we encode the preferences with the probability distributions of word labels, because the label of a word indicates its entity type and relative position from the entities (e.g., *B-ORG* means the first word of an organization). To learn the distributions automatically, we propose an auxiliary word classification task and employ a two-layer neural network as the classifier. The classifier’s input consists of the morphological representation  $\mathbf{c}_k$  and the word embedding  $\mathbf{w}_k$ . Besides, we add a position embedding  $\mathbf{p}_k$  to represent the relative position information:

$$\begin{aligned} \mathbf{p}_k &= E(k; \theta_p) \in \mathbb{R}^{d_{pe}} \\ \mathbf{x}'_k &= [\mathbf{c}_k; \mathbf{w}_k; \mathbf{p}_k] \in \mathbb{R}^{d_{we} + d_{ch} + d_{pe}} \end{aligned} \quad (8)$$

where  $\mathbf{p}_k$  is obtained by looking up a randomly-initialized embedding matrix and tuned during training. Then  $\mathbf{x}'_k$  is fed into the two-layer classifier to predict label distribution:

$$\begin{aligned} \mathbf{m}_k &= \tanh(\mathbf{W}_{c_1} \mathbf{x}'_k + \mathbf{b}_{c_1}) \\ \mathbf{p}_k^{label} &= \text{softmax}(\mathbf{W}_{c_2} \mathbf{m}_k + \mathbf{b}_{c_2}) \end{aligned} \quad (9)$$

where  $\mathbf{W}_{c_1} \in \mathbb{R}^{d_{lch} \times (d_{we} + d_{ch} + d_{pe})}$ ,  $\mathbf{b}_{c_1} \in \mathbb{R}^{d_{lch}}$ ,  $\mathbf{W}_{c_2} \in \mathbb{R}^{|C| \times d_{lch}}$ ,  $\mathbf{b}_{c_2} \in \mathbb{R}^{|C|}$  are the parameters of the classifier (the number of all labels denoted as  $|C|$ ). During training, we use  $\mathbf{p}_k^{label}$  to compute the loss function for word classification, which is formulated as cross-entropy loss:

$$L_{WC}(\theta) = - \sum_{k=1}^n \log p_k^{label}(y_k | \theta). \quad (10)$$

To incorporate the word-level context, we concatenate  $\mathbf{p}_k^{label}$  with the original CRF input  $\mathbf{h}_{ik}^{seq}$  to enrich word representations with the label distributions (Seyler et al., 2018). The CRF takes the enhanced word representations as input and decodes the best label sequence. Our framework is jointly trained on the original NER and the auxiliary classification task via multi-task learning:

$$L(\theta) = L_{CRF}(\theta) + \lambda L_{WC}(\theta), \quad (11)$$

where  $\lambda$  is the weight of word classification loss.

## 4 Experiments

### 4.1 Datasets and Evaluation Metrics

We evaluate our approach on the CoNLL-2002, CoNLL-2003, and Wikigold NER datasets. The Wikigold dataset contains annotations for English (denoted as **WIKI**). The CoNLL-2002 dataset contains annotations for Dutch (denoted as **NLD**)<sup>1</sup>. The CoNLL-2003 dataset contains annotations for English and German (denoted as **ENG** and **DEU** respectively). All datasets are manually tagged with four different entity types (*LOC*, *PER*, *ORG*, *MISC*). The CoNLL datasets have standard train, development, and test sets. Since the Wikigold dataset doesn’t have standard separation, we randomly split the data into the three sets and perform all experiments on the same separation. Table 1 shows the number of documents and sentences of the datasets. We report the official micro-averaged  $F_1$  scores on all the datasets.

Dataset	Train	Dev.	Test
WIKI	101 (1,227)	22 (402)	22 (212)
NLD	287 (16,093)	74 (2,969)	119 (5,314)
DEU	533 (12,705)	201 (3,068)	155 (3,160)
ENG	946 (14,987)	216 (3,466)	231 (3,684)

Table 1: Numbers of documents (and sentences) in datasets statistics.

### 4.2 Experimental Settings

In our experiments, we use the BIOES labeling scheme for output tags, which was proven to outperform other options in previous work (Ratinov and Roth, 2009). Under this tagging scheme, the number of labels  $|C| = 17$  ( $[B,I,E,S] \times$

<sup>1</sup>The CoNLL-2002 dataset contains Dutch and Spanish data. But the Spanish data lacks the marks of document boundaries. Thus we only conduct experiments on the Dutch data.

Hyper-parameter	Value
Word embedding dim. ( $d_{we}$ )	50/300
Character embedding dim. ( $d_{ce}$ )	25
Position embedding dim. ( $d_{pe}$ )	30
Character hidden state dim. ( $d_{ch}$ )	80
Word hidden state dim. ( $d_{wh}$ )	300
Sentence hidden state dim. ( $d_{sh}$ )	300
Sequence hidden state dim. ( $d_{sqh}$ )	300
Neural attention subspace dim. ( $d_{na}$ )	100
Self attention subspace dim. ( $d_{sa}$ )	60
Label classifier hidden dim. ( $d_{lch}$ )	64
Number of heads ( $H$ )	5
Weight of $L_{WC}$ ( $\lambda$ )	0.1

Table 2: Hyper-parameters of our model.

[LOC, PER, ORG, MISC] +  $O$ ). For English datasets, we use the 50-dimensional Senna word embeddings (Collobert et al., 2011) and pre-process the text by lower-casing the words and replacing all digits with 0 (Chiu and Nichols, 2016; Peters et al., 2017). For Dutch and German datasets, we use the pre-trained 300-dimensional word2vec embeddings (Mikolov et al., 2013), which are trained on the Wikipedia dumps<sup>2</sup>. We adopt ELMo (Peters et al., 2018; Che et al., 2018) as the pre-trained LM embeddings<sup>3</sup>. The hyper-parameters of our model are shown in Table 2. For regularization, we add 25% dropout (Srivastava et al., 2014) to the input of all BiGRUs, but not to the recurrent connections.

Following Peters et al. (2017), we use the Adam optimizer (Kingma and Ba, 2014) with gradient norms clipped at 5.0. We fine-tune the pre-trained word embeddings and ELMo model parameters. We train our model with a constant learning rate of  $\gamma = 0.001$  for 20 epochs. Then we start a simple learning rate decay schedule: divide  $\gamma$  by ten, train for 5 epochs, divide  $\gamma$  by ten, train for 5 epochs again and stop. We train the model’s parameters on the train set and tune the hyper-parameters on the development set. Then we compute  $F_1$  score on the test set at the epoch with the highest development performance. Following previous work (Chiu and Nichols, 2016; Peters et al., 2017), we train our model for multiple times with different random

seeds and report the mean of  $F_1$ .

### 4.3 Performance Evaluation

To demonstrate the effectiveness of our method, we compare our experimental results on the CoNLL-2002 and CoNLL-2003 datasets with previously published state-of-the-art models: Ando and Zhang (2005) proposed a structural learning algorithm for semi-supervised NER; Qi et al. (2009) proposed Word-Class Distribution Learning (WCDL) method; Nothman et al. (2013) introduced Wikipedia articles as extra knowledge; Gillick et al. (2015) proposed a byte-level model for multilingual NER; Lample et al. (2016) proposed BiLSTM-CRF model; Yang et al. (2017) applied transfer learning mechanism for NER; Peters et al. (2018) proposed ELMo embeddings; Clark et al. (2018) proposed Cross-View Training (CVT) method; Devlin et al. (2019) proposed BERT representations; Liu et al. (2019) introduced external gazetteers to this task; Akbik et al. (2018) proposed contextual character language model and achieved the state-of-the-art performance; Zhang et al. (2018) and Hu et al. (2020) utilized global attention to mine document-level information; Gui et al. (2020) used memory mechanism to capture document-level label consistency. Table 3 shows the comparison results, from which we can observe that the incorporation of multi-level contexts brings 0.47%,

Method	ENG	DEU	NLD
Ando et al. (2005)	89.31	75.27	—
Qi et al. (2009)	88.69	75.72	—
Nothman et al. (2013)	85.2	66.5	78.6
Gillick et al. (2015)	86.50	76.22	82.84
Lample et al. (2016)	90.94	78.76	81.74
Yang et al. (2017)	91.26	—	85.19
Peters et al. (2018)	92.22	—	—
Clark et al. (2018)	92.6	—	—
Akbik et al. (2018)	<b>93.09</b>	<b>88.32</b>	—
Devlin et al. (2019)	92.8	—	—
Liu et al. (2019)	92.75	—	—
Zhang et al. (2018)	91.81	79.21	87.40
Hu et al. (2020)	91.92	—	—
Gui et al. (2020)	93.05	—	—
TagLM (Peters et al., 2017)	91.93	—	—
TagLM+ELMo (baseline)	92.21 <sup>†</sup>	77.83 <sup>†</sup>	88.05 <sup>†</sup>
Our model	<b>92.68*</b>	<b>78.87*</b>	<b>88.93*</b>

Table 3: Comparison results of  $F_1$  score on the CoNLL-2002 and CoNLL-2003 test sets. <sup>†</sup> denotes the results of our implementation. \* denotes statistically significant improvements over the baseline model with  $p < 0.01$  under a  $t$ -test.

<sup>2</sup><https://github.com/Kyubyong/wordvectors>

<sup>3</sup>We also conduct experiments with TagLM+BERT<sub>BASE</sub> with released parameters. Due to the limitation of GPU memory, we didn’t fine-tune BERT. The dev and test set  $F_1$  scores are **95.03±0.22** and **91.64±0.18** respectively. Our results have a surprisingly huge gap between the reported scores (we refer readers to Section 4.3 and 5.4 of Devlin et al. (2019)).

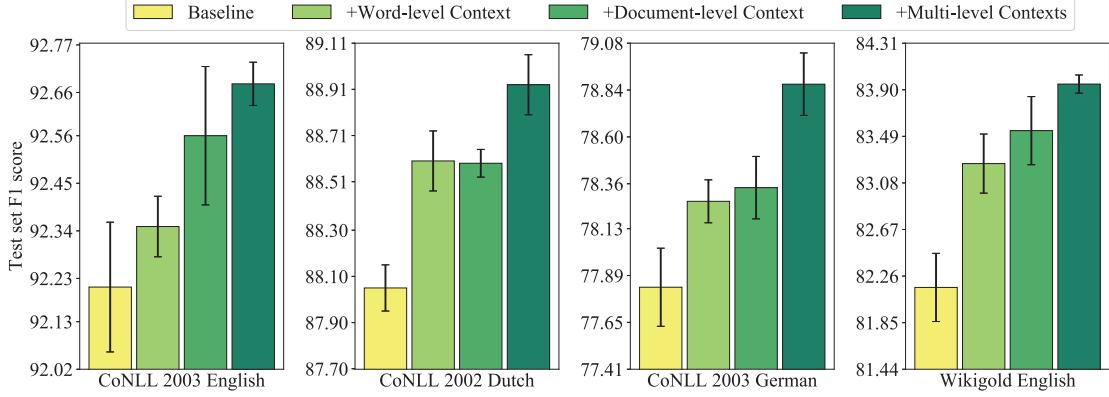


Figure 3: An ablation study of our framework. We compare the mean of test set  $F_1$  score under the four settings on the four datasets. The bars indicate the standard deviation of  $F_1$  score.

1.04%, and 0.88% absolute  $F_1$  score improvement on the English, German and Dutch dataset respectively compared with our baseline model. In addition, our model outperforms most of the previous sentence- and document-level methods on the three languages. The improvements demonstrate the effectiveness of our framework, which fully exploits the document and word-level contexts and combines the multi-level contexts. With the assistance of multi-level contexts, our model can capture more contextual information beyond sentences and recognize entities more correctly.

#### 4.4 Ablation Study

To study the contribution of the document- and word-level context respectively, we conduct experiments on two settings: only incorporating the word-level context and the document-level context, and compare the  $F_1$  score with our model. Figure 3 shows the results, from which we have the following observations: (1) The document- and word-level contexts both bring improvements on the four datasets. It indicates the utility of these contexts respectively. The document-level context contains interactions between sentences within a document. The word-level context contains words’ type and position preferences. Either of the contexts can help alleviate the effects of limited or ambiguous sentence context. (2) The multi-level contexts method improves the  $F_1$  score over the other two settings on all the datasets. It validates the effectiveness of the fusion of multi-level contexts. Our framework can exploit and fuse the contexts at the document and word level simultaneously. With the assistance of more extra contextual information from the document and word level, our

method performs better than the other two settings of combining only one context.

### 4.5 Analysis

#### 4.5.1 How to fuse the document-level context?

In this experiment, we propose four alternative ways to fuse document-level contextual representations  $\mathbf{d}_i$  with sentence-level contextual representations  $\mathbf{h}_i^{word}$  or  $\mathbf{h}_i^{seq}$  (Equation 2):

- Concatenate  $\mathbf{h}_{ik}^{word}$  with  $\mathbf{d}_i$ ;
- Add  $\mathbf{h}_{ik}^{word}$  to  $\mathbf{d}_i$ ;
- Initialize  $\mathbf{h}_{i(-1)}^{seq}$  with  $\mathbf{d}_i$ ;
- Replace  $\mathbf{h}_{i0}^{word}$  with  $\mathbf{d}_i$ .

Table 4 shows the comparison result on the CoNLL-2003 English test set. The first two options essentially translate  $\mathbf{h}_{ik}^{word}$  in the vector space, because they enhance  $\mathbf{h}_{ik}^{word}$  with the same  $\mathbf{d}_i$  for all words. Therefore they cannot fully combine the contexts. To distinguish between the latter two options, we need to focus on the internal calculation of GRU:  $\mathbf{h}_t = (1 - z_t)\mathbf{n}_t + z_t\mathbf{h}_{t-1}, \mathbf{n}_t = \tanh(\mathbf{W}_{in}\mathbf{x}_t + \mathbf{b}_{in} + r_t(\mathbf{W}_{hn}\mathbf{h}_{t-1} + \mathbf{b}_{hn}))$ . GRU uses non-linearly transformed  $\mathbf{x}_t$  and raw  $\mathbf{h}_t$  to calculate hidden states. We speculate that the non-linear transformation on  $\mathbf{d}_i$  aligns it to the same space as  $\mathbf{h}_{ik}^{word}$  and produces better performance.

#### 4.5.2 How to fuse the word-level context?

In this experiment, we compare three ways of fusing word-level contextual representations  $\mathbf{p}_i^{label}$  with the sentence-level context:

- Concatenate the input  $\mathbf{x}_k$  with  $\mathbf{p}_i^{label}$ ;
- Concatenate  $\mathbf{h}_{ik}^{word}$  with  $\mathbf{p}_i^{label}$ ;
- Concatenate  $\mathbf{h}_{ik}^{seq}$  with  $\mathbf{p}_i^{label}$ .

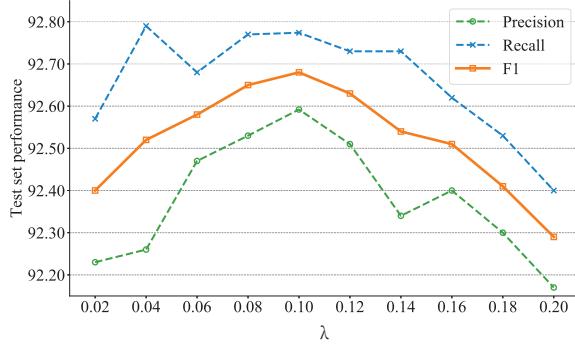


Figure 4: The CoNLL 2003 English test set performance of our model with different  $\lambda$ .

Table 5 shows the comparison results. The first two options use BiGRU to encode the label distributions but perform worse than the last one using CRF. We speculate that CRF is more suitable to encode the distributions of word label than BiGRU because there exist strong connections between two adjacent words’ label distributions intuitively.

#### 4.5.3 Which attention mechanism to use at document level?

In this part, we compare three choices of attention mechanism: the *multi-head self attention*, *self attention*, and the most-popular *neural attention* mechanism. Table 6 shows the comparison results. We can observe that the self attention mechanism outperforms neural attention because it can capture interactions between sentences in the document. In contrast, the neural attention mechanism only learns the sentence’s weight based on its representation, thus fails to capture the interactions. Furthermore, multi-head self attention performs better than self attention because it can capture a sentence’s interactions with multiple sentences.

#### 4.5.4 How to choose the weight $\lambda$ of the auxiliary task ?

We conduct experiments on different weights  $\lambda$  to investigate its influence and illustrate the result in Figure 4. We speculate that  $\lambda$  controls the propor-

Document-level fusion method	$F_1 \pm \text{std}$
Concatenate $\mathbf{h}_{ik}^{word}$ with $\mathbf{d}_i$	$92.36 \pm 0.08$
Add $\mathbf{h}_{ik}^{word}$ to $\mathbf{d}_i$	$92.43 \pm 0.05$
Initialize $\mathbf{h}_{i(-1)}^{seq}$ with $\mathbf{d}_i$	$92.42 \pm 0.10$
Replace $\mathbf{h}_{i0}^{word}$ with $\mathbf{d}_i$	<b><math>92.68 \pm 0.09</math></b>

Table 4: Comparison of different ways of fusing the document-level context on CoNLL 2003 test set.

Fusion method	$F_1 \pm \text{std}$
Concatenate $\mathbf{p}_{ik}^{label}$ with $\mathbf{x}_k$	$91.99 \pm 0.14$
Concatenate $\mathbf{p}_{ik}^{label}$ with $\mathbf{h}_{ik}^{word}$	$92.33 \pm 0.11$
Concatenate $\mathbf{p}_{ik}^{label}$ with $\mathbf{h}_{ik}^{seq}$	<b><math>92.68 \pm 0.09</math></b>

Table 5: Comparison of different ways of fusing the word-level context on CoNLL 2003 test set.

Attention mechanism	$F_1 \pm \text{std}$
Neural attention	$92.49 \pm 0.10$
Self attention	$92.52 \pm 0.09$
Multi-head self attention	<b><math>92.68 \pm 0.09</math></b>

Table 6: Comparison of different attention mechanisms at document level on CoNLL 2003 test set.

Case #1	Label	LITTLE TO MISS CAMPESSE FAREWELL
	TagLM	LITTLE TO MISS CAMPESSE FAREWELL
	Ours	LITTLE TO MISS CAMPESSE FAREWELL
	D-lvl	Centre Jason Little will miss ...
Case #2	Label	... play at the Melbourne Cricket Ground.
	TagLM	... play at the Melbourne Cricket Ground.
	Ours	... play at the Melbourne Cricket Ground.
	W-lvl	1. ... the Sydney Cricket Ground ... 2. ... the Melbourne Cricket Ground ...

Table 7: Comparison between the baseline and our method on two cases. Blue, red and orange entities indicate the names of organizations, persons and locations. The bold words are word-level (W-lvl) or document-level (D-lvl) supporting contextual evidence.

tion of the word-level context in all contexts. When  $\lambda$  changes, the balance of the contexts is broken, and the performance is affected. Besides,  $\lambda$  controls the learning rate of the word label classifier’s parameters. Its increase and decrease will hurt the accuracy of the label classification.

#### 4.6 Case Study

Table 7 shows the comparison of the baseline and our model on two example sentences. In the first case, the ambiguity of *LITTLE* disturbs the baseline model. Our model finds another explicit mention *Jason Little* as a person (centre) in this document and correctly identifies this entity. In the second case, the *Melbourne Cricket Ground* (location) is wrongly classified as organization, because one can either *play* at a team or *play* at a stadium. Our model notices the two other mentions of *Ground*, both of which appears as the last word of location, and corrects the erroneous entity type. The examples prove that our model can mine contextual information outside sentences and recognize

entities more correctly than the baseline model.

## 5 Conclusion

In this paper, we propose a unified structure to incorporate multi-level contexts for the NER task. We use TagLM as our baseline model to capture the sentence-level context. To incorporate the document-level context, we propose to learn relationships between sentences within a document with the multi-head self attention mechanism. Besides, to mine word-level contextual information, we propose an auxiliary task to predict the word type to capture its type preferences. Our model is jointly trained on the NER and auxiliary tasks through multi-task learning. We evaluate our model on several benchmark datasets, and the experimental results prove the effectiveness of our method.

## Acknowledgement

This work was supported by the National Key Research and Development Program of China under Grant number 2018YFB2101501, the National Natural Science Foundation of China under Grant numbers U1936208, U1936216.

## References

- Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649.
- Rie Kubota Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *JMLR*, 6(Nov):1817–1853.
- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.
- Wanxiang Che, Yijia Liu, Yuxuan Wang, Bo Zheng, and Ting Liu. 2018. Towards better ud parsing: Deep contextualized word embeddings, ensemble, and treebank concatenation. In *CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 55–64.
- Jason PC Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional lstm-cnns. *TACL*, 4:357–370.
- Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Workshop on Deep Learning, December 2014*.
- Kevin Clark, Minh-Thang Luong, Christopher D Manning, and Quoc Le. 2018. Semi-supervised sequence modeling with cross-view training. In *EMNLP*, pages 1914–1925.
- Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *JMLR*, 12(Aug):2493–2537.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. pages 4171–4186.
- Ronen Feldman and Benjamin Rosenfeld. 2006. Boosting unsupervised relation extraction by using ner. In *EMNLP*, pages 473–481.
- Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. 2000. Learning to forget: Continual prediction with lstm. *Neural Computation*, 12(10):2451–2471.
- Dan Gillick, Cliff Brunk, Oriol Vinyals, and Amarnag Subramanya. 2015. Multilingual language processing from bytes. *arXiv preprint arXiv:1512.00103*.
- Tao Gui, Jiacheng Ye, Qi Zhang, Yaqian Zhou, Yeyun Gong, and Xuanjing Huang. 2020. Leveraging document-level label consistency for named entity recognition. In *IJCAI*, pages 3976–3982.
- Anwen Hu, Zhicheng Dou, Jian-Yun Nie, and Ji-Rong Wen. 2020. Leveraging multi-token entities in document-level named entity recognition. In *AAAI*, pages 7961–7968.
- Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- John D Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, pages 282–289.
- Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *NAACL-HLT*, pages 260–270.
- Changki Lee, Yi-Gyu Hwang, Hyo-Jung Oh, Soojong Lim, Jeong Heo, Chung-Hee Lee, Hyeyon-Jin Kim, Ji-Hyun Wang, and Myung-Gil Jang. 2006. Fine-grained named entity recognition using conditional random fields for question answering. In *Asia Information Retrieval Symposium*, pages 581–587. Springer.
- Dekang Lin and Xiaoyun Wu. 2009. Phrase clustering for discriminative learning. In *ACL-AFNLP*, pages 1030–1038. Association for Computational Linguistics.
- Tianyu Liu, Jin-Ge Yao, and Chin-Yew Lin. 2019. Towards improving neural named entity recognition with gazetteers. In *ACL*, pages 5301–5307.

- Gang Luo, Xiaojiang Huang, Chin-Yew Lin, and Zaqing Nie. 2015. Joint entity recognition and disambiguation. In *EMNLP*, pages 879–888.
- Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *ACL*, volume 1, pages 1064–1074.
- Andrei Mikheev, Marc Moens, and Claire Grover. 1999. Named entity recognition without gazetteers. In *EACL*, pages 1–8.
- Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.
- Joel Nothman, Nicky Ringland, Will Radford, Tara Murphy, and James R Curran. 2013. Learning multilingual named entity recognition from wikipedia. *Artificial Intelligence*, 194:151–175.
- Alexandre Passos, Vineet Kumar, and Andrew McCallum. 2014. Lexicon infused phrase embeddings for named entity resolution. In *CoNLL*, pages 78–86.
- Matthew Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. 2017. Semi-supervised sequence tagging with bidirectional language models. In *ACL*, volume 1, pages 1756–1765.
- Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL-HLT*, volume 1, pages 2227–2237.
- Yanjun Qi, Ronan Collobert, Pavel Kuksa, Koray Kavukcuoglu, and Jason Weston. 2009. Combining labeled and unlabeled data with word-class distribution learning. In *CIKM*, pages 1737–1740. ACM.
- Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *CoNLL*, pages 147–155.
- Dominic Seyler, Tatiana Dembelova, Luciano Del Corro, Johannes Hoffart, and Gerhard Weikum. 2018. A study of the importance of external knowledge in the named entity recognition task. In *ACL*, volume 2, pages 241–246.
- Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, 15(1):1929–1958.
- Emma Strubell, Patrick Verga, David Belanger, and Andrew McCallum. 2017. Fast and accurate entity recognition with iterated dilated convolutions. In *EMNLP*, pages 2670–2680.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*, pages 5998–6008.
- Guohai Xu, Chengyu Wang, and Xiaofeng He. 2018. Improving clinical named entity recognition with global neural attention. In *Asia-Pacific Web (APWeb) and Web-Age Information Management (WAIM) Joint International Conference on Web and Big Data*, pages 264–279. Springer.
- Zhilin Yang, Ruslan Salakhutdinov, and William W Cohen. 2017. Transfer learning for sequence tagging with hierarchical recurrent networks. *arXiv preprint arXiv:1703.06345*.
- Boliang Zhang, Spencer Whitehead, Lifu Huang, and Heng Ji. 2018. Global attention for name tagging. In *CoNLL*, pages 86–96.

# A General Framework for Adaptation of Neural Machine Translation to Simultaneous Translation

Yun Chen<sup>\*1</sup>, Liangyou Li<sup>2</sup>, Xin Jiang<sup>2</sup>, Xiao Chen<sup>2</sup>, Qun Liu<sup>2</sup>

<sup>1</sup>Shanghai University of Finance and Economics, Shanghai, China

<sup>2</sup>Huawei Noah’s Ark Lab, Hong Kong, China

yunchen@sufe.edu.cn, {liliangyou, jiang.xin, chen.xiao2, qun.liu}@huawei.com

## Abstract

Despite the success of neural machine translation (NMT), simultaneous neural machine translation (SNMT), the task of translating in real time before a full sentence has been observed, remains challenging due to the syntactic structure difference and simultaneity requirements. In this paper, we propose a general framework for adapting neural machine translation to translate simultaneously. Our framework contains two parts: prefix translation that utilizes a consecutive NMT model to translate source prefixes and a stopping criterion that determines when to stop the prefix translation. Experiments on three translation corpora and two language pairs show the efficacy of the proposed framework on balancing the quality and latency in adapting NMT to perform simultaneous translation.

## 1 Introduction

Simultaneous translation (Fügen et al., 2007; Oda et al., 2014; Grissom et al., 2014; Niehues et al., 2016; Cho and Esipova, 2016; Gu et al., 2017; Ma et al., 2018), the task of producing a partial translation of a sentence before the whole input sentence ends, is useful in many scenarios including outbound tourism, international summit and multilateral negotiations. Different from the consecutive translation in which translation quality alone matters, simultaneous translation trades off between translation quality and latency. The syntactic structure difference between the source and target language makes simultaneous translation more challenging. For example, when translating from a verb-final (SOV) language (e.g., Japanese) to a verb-media (SVO) language (e.g., English), the verb appears much later in the source sequence

than in the target language. Some premature translations can lead to significant loss in quality (Ma et al., 2018).

Recently, a number of researchers have endeavored to explore methods for simultaneous translation in the context of NMT (Bahdanau et al., 2015; Vaswani et al., 2017). Some of them propose sophisticated training frameworks explicitly designed for simultaneous translation (Ma et al., 2018; Arivazhagan et al., 2019). These approaches are either memory inefficient during training (Ma et al., 2018) or with hyper-parameters hard to tune (Arivazhagan et al., 2019). Others utilize a full-sentence base model to perform simultaneous translation by modifications to the encoder and the decoding process. To match the incremental source context, they replace the bidirectional encoder with a left-to-right encoder (Cho and Esipova, 2016; Satija and Pineau, 2016; Gu et al., 2017; Alinejad et al., 2018) or recompute the encoder hidden states (Zheng et al., 2019). On top of that, heuristic algorithms (Cho and Esipova, 2016; Dalvi et al., 2018) or a READ/WRITE model trained with reinforcement learning (Satija and Pineau, 2016; Gu et al., 2017; Alinejad et al., 2018) or supervised learning (Zheng et al., 2019) are used to decide, at every step, whether to wait for the next source token or output a target token. However, these models either cannot directly use a pretrained consecutive neural machine translation (CNMT) model with bidirectional encoder as the base model or work in a sub-optimal way in the decoding stage.

In this paper, we study the problem of adapting neural machine translation to translate simultaneously. We formulate simultaneous translation as two nested loops: an outer loop that updates input buffer with newly observed source tokens and an inner loop that translates source tokens in the buffer updated at each outer step. For the outer loop, the input buffer can be updated by an ASR system with

<sup>\*</sup>Part of the work was done when Yun is working at Huawei Noah’s Ark Lab.

an arbitrary update schedule. For the inner loop, we translate using the pretrained CNMT model and stop translation with a stopping controller. Such formulation is different from previous work (Satija and Pineau, 2016; Gu et al., 2017; Alinejad et al., 2018; Zheng et al., 2019) which define simultaneous translation as sequentially making interleaved READ or WRITE decisions. We argue that our formulation is better than the previous one in two aspects: (i) Our formulation can better utilize the available source tokens. Under previous formulation, the number of source tokens observed by the CNMT model is determined by the number of READ actions that has been produced by the policy network. It is likely that the CNMT model does not observe all the available source tokens produced by the ASR system. In contrast, the CNMT model observes all the available source tokens when performing inner loop translation in our framework. (ii) Previous formulation makes  $T_\eta + T_\tau$  READ or WRITE decisions regardless of the ASR update schedule, where  $T_\eta$  and  $T_\tau$  are source sentence and translation length, respectively. For an ASR system that outputs multiple tokens at a time, this is computational costly. Consider an extreme case where the ASR system outputs a full source sentence at a time. Previous work translates with a sequence of  $T_\eta + T_\tau$  actions, while we translate with a sequence of  $T_\tau$  decisions ( $T_\tau - 1$  CONTINUE and 1 STOP).

Under our proposed framework, we present two schedules for simultaneous translation: one stops the inner loop translation with heuristic and one with a stopping controller learned in a reinforcement learning framework to balance translation quality and latency. We evaluate our method on IWSLT16 German-English (DE-EN) translation in both directions, WMT15 English-German (EN-DE) translation in both directions, and NIST Chinese-to-English (ZH→EN) translation. The results show our method with reinforced stopping controller consistently improves over the de-facto baselines, and achieves low latency and reasonable BLEU scores.

## 2 Background

Given a set of source-target sentence pairs  $\langle \mathbf{x}_m, \mathbf{y}_m^* \rangle_{m=1}^M$ , a consecutive NMT model can be trained by maximizing the log-likelihood of the target sentence from its entire source side context:

$$\hat{\phi} = \operatorname{argmax}_{\phi} \left\{ \sum_{m=1}^M \log p(\mathbf{y}_m^* | \mathbf{x}_m; \phi) \right\}, \quad (1)$$

where  $\phi$  is a set of model parameters. At inference time, the NMT model first encodes a source language sentence  $\mathbf{x} = \{x_1, \dots, x_{T_\eta}\}$  with its encoder and passes the encoded representations  $\mathbf{h} = \{h_1, \dots, h_{T_\eta}\}$  to a greedy decoder. Then the greedy decoder generates a translated sentence in target language by sequentially choosing the most likely token at each step  $t$ :

$$y_t = \operatorname{argmax}_y p(y | y_{<t}, \mathbf{x}). \quad (2)$$

The distribution of next target word is defined as:

$$\begin{aligned} p(y | y_{<t}, \mathbf{x}) &\propto \exp [\phi_{\text{OUT}}(z_t)] \\ z_t &= \phi_{\text{DEC}}(y_{t-1}, z_{<t}, \mathbf{h}), \end{aligned} \quad (3)$$

where  $z_t$  is the decoder hidden state at position  $t$ . In consecutive NMT, once obtained, the encoder hidden states  $\mathbf{h}$  and the decoder hidden state  $z_t$  are not updated anymore and will be reused during the entire decoding process.

## 3 Simultaneous NMT

In SNMT, we receive streaming input tokens, and learn to translate them in real-time. We formulate simultaneous translation as two nested loops: the outer loop that updates an input buffer with newly observed source tokens and the inner loop that translates source tokens in the buffer updated at each outer step.

More precisely, suppose at the end of outer step  $s - 1$ , the input buffer is  $\mathbf{x}^{s-1} = \{x_1, \dots, x_{\eta[s-1]}\}$ , and the output buffer is  $\mathbf{y}^{s-1} = \{y_1, \dots, y_{\tau[s-1]}\}$ . Then at outer step  $s$ , the system translates with the following steps:

- 1 The system observes  $c_s > 0$  new source tokens and updates the input buffer to be  $\mathbf{x}^s = \{x_1, \dots, x_{\eta[s]}\}$  where  $\eta[s] = \eta[s-1] + c_s$ .
- 2 Then, the system starts inner loop translation and writes  $w_s >= 0$  target tokens to the output buffer. The output buffer is updated to be  $\mathbf{y}^s = \{y_1, \dots, y_{\tau[s]}\}$  where  $\tau[s] = \tau[s-1] + w_s$ .

The simultaneous decoding process continues until no more source tokens are added in the outer loop. We define the last outer step as the terminal outer step  $S$ , and other outer steps as non-terminal outer steps.

For the outer loop, we make no assumption about the value of  $c_s$ , while all previous work assumes

$c_s = 1$ . This setting is more realistic because (i) increasing  $c_s$  can reduce the number of outer steps, thus reducing computation cost; (ii) in a real speech translation application, an ASR system may generate multiple tokens at a time.

For the inner loop, we adapt a pretrained vanilla CNMT model to perform partial translation with two important concerns:

1. Prefix translation: given a source prefix  $\mathbf{x}^s = \{x_1, \dots, x_{\eta[s]}\}$  and a target prefix  $\mathbf{y}_{\tau[s-1]}^s = \{y_1, \dots, y_{\tau[s-1]}\}$ , how to predict the remaining target tokens?
2. Stopping criterion: since the NMT model is trained with full sentences, how to design the stopping criterion for it when translating partial source sentences?

### 3.1 Prefix Translation

At an outer step  $s$ , given encoder hidden states  $\mathbf{h}^s$  for source prefix  $\mathbf{x}^s = \{x_1, \dots, x_{\eta[s]}\}$  and decoder hidden states  $\mathbf{z}_{\tau[s-1]}^s$  for target prefix  $\mathbf{y}_{\tau[s-1]}^s = \{y_1, \dots, y_{\tau[s-1]}\}$ , we perform prefix translation sequentially with a greedy decoder:

$$\begin{aligned} z_t^s &= \phi_{\text{DEC}}(y_{t-1}, z_{<t}^s, \mathbf{h}^s) \\ p(y|y_{<t}, \mathbf{x}^s) &\propto \exp[\phi_{\text{OUT}}(z_t^s)] \\ y_t &= \text{argmax}_y p(y|y_{<t}, \mathbf{x}^s), \end{aligned} \quad (4)$$

where  $t$  starts from  $t = \tau[s-1] + 1$ . The prefix translation terminates when a stopping criterion meets, yielding a translation  $\mathbf{y}^s = \{y_1, \dots, y_{\tau[s]}\}$ .

However, a major problem comes from the above translation method: how can we obtain the encoder hidden states  $\mathbf{h}^s$  and decoder hidden states  $\mathbf{z}_{\tau[s-1]}^s$  at the beginning of prefix translation? We propose to rebuild all encoder and decoder hidden states with

$$\mathbf{h}^s = \phi_{\text{ENC}}(\mathbf{x}^s), \quad (5)$$

$$\mathbf{z}_{\tau[s-1]}^s = \phi_{\text{DEC}}(\mathbf{y}_{\tau[s-1]}^s, \mathbf{h}^s). \quad (6)$$

During full sentence training, all the decoder hidden states are computed conditional on the same source tokens. By rebuilding encoder and decoder hidden states, we also ensure that the decoder hidden states are computed conditional on the same source. This strategy is different from previous work that reuse previous encoder (Cho and Esipova, 2016; Gu et al., 2017; Dalvi et al., 2018; Alinejad et al., 2018) or decoder (Cho and Esipova, 2016; Gu et al., 2017; Dalvi et al., 2018; Ma et al., 2018)

	src	trans
1	晓莹	xiaoying
2	晓莹 你	xiaoying you
3	晓莹 你好	xiaoying you are good
4	晓莹 你好。	xiaoying you are good .

Figure 1: Failure case when using EOS alone as the stopping criterion.

hidden states. We carefully compare the effect of rebuilding hidden states in Section 4.2 and experiment results show that rebuilding all hidden states benefits translation.

### 3.2 Stopping Criterion

In consecutive NMT, the decoding algorithm such as greedy decoding or beam search terminates when the translator predicts an EOS token or the length of the translation meets a predefined threshold (e.g. 200). The decoding for most source sentences terminates when the translator predicts the EOS token.<sup>1</sup> In simultaneous decoding, since we use a NMT model pretrained on full sentences to translate partial source sentences, it tends to predict EOS when the source context has been fully translated. However, such strategy could be too aggressive for simultaneous translation. Fig. 1 shows such an example. At outer step 2, the translator predicts “you EOS”, emitting target token “you”. However, “you” is not the expected translation for “你” in the context of “你好。”. Therefore, we hope prefix translation at outer step 2 can terminate without emitting any words.

To alleviate such problems and do better simultaneous translation with pretrained CNMT model, we propose two novel stopping criteria for prefix translation.

#### 3.2.1 Length and EOS Control

In consecutive translation, the decoding process stops mainly when predicting EOS. In contrast, for prefix translation at non-terminal outer step, we stop the translation process when translation length is  $d$  tokens behind source sentence length:  $\tau[s] = \eta[s] - d$ . Specifically, at the beginning of outer step  $s$ , we have source prefix  $\mathbf{x}^s = \{x_1, \dots, x_{\eta[s]}\}$  and target prefix  $\mathbf{y}_{\tau[s-1]}^s = \{y_1, \dots, y_{\tau[s-1]}\}$ . Prefix translation terminates at inner step  $w_s$  when

<sup>1</sup>We conduct greedy decoding on the validation set of WMT15 EN→DE translation with fairseq-py, and find that 100% translation terminates with EOS predicted.

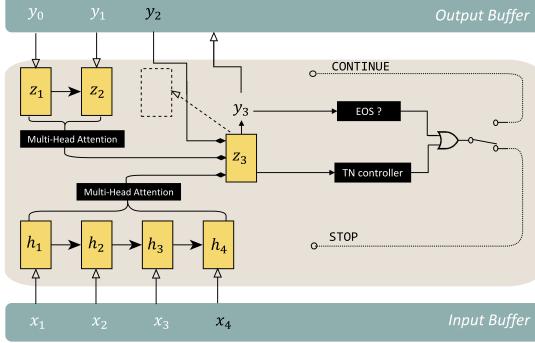


Figure 2: Framework of our proposed model with the TN controller.

predicting an EOS token or satisfying:

$$w_s = \begin{cases} \max(0, \eta[s] - \tau[s-1] - d) & s < S \\ 200 - \tau[s-1] & s = S \end{cases} \quad (7)$$

where  $d$  is a non-negative integer that determines the translation latency of the system. We call this stopping criterion as Length and EOS (LE) stopping controller.

### 3.2.2 Learning When to Stop

Although simple and easy to implement, LE controller lacks the capability to learn the optimal timing with which to stop prefix translation. Therefore, we design a small trainable network called trainable (TN) stopping controller to learn when to stop prefix translation for non-terminal outer step. Fig. 2 shows the illustration.

At each inner decoding step  $k$  for non-terminal outer step  $s$ , the TN controller utilizes a stochastic policy  $\pi_\theta$  parameterized by a neural network to make the binary decision on whether to stop translation at current step:

$$\pi_\theta(a_{\tau[s-1]+k} | z_{\tau[s-1]+k}^s) = f_\theta(z_{\tau[s-1]+k}^s), \quad (8)$$

where  $z_{\tau[s-1]+k}^s$  is the current decoder hidden state. We implement  $f_\theta$  with a feedforward network with two hidden layers, followed by a softmax layer. The prefix translation stops if the TN controller predicts  $a_{\tau[s-1]+k} = 1$ . Our TN controller is much simpler than previous work (Gu et al., 2017) which implements the READ/WRITE policy network using a recurrent neural network whose input is the combination of the current context vector, the current decoder state and the embedding vector of the candidate word.

To train the TN controller, we freeze the NMT model with pretrained parameters, and optimize

the TN network with policy gradient for reward maximization  $\mathcal{J} = \mathbb{E}_{\pi_\theta}(\sum_{t=1}^{T_\tau} r_t)$ . With a trained TN controller, prefix translation stops at inner decoding step  $w_s$  when predicting an EOS token or satisfying:

$$\begin{cases} a_{\tau[s-1]+w_s} = 1 & s < S \\ w_s = 200 - \tau[s-1] & s \leq S \end{cases}. \quad (9)$$

In the following, we talk about the details of the reward function and the training with policy gradient.

**Reward** To trade-off between translation quality and latency, we define the reward function at inner decoding step  $k$  of outer step  $s$  as:

$$r_t = r_t^Q + \alpha \cdot r_t^D, \quad (10)$$

where  $t = \tau[s-1]+k$ , and  $r_t^Q$  and  $r_t^D$  are rewards related to quality and delay, respectively.  $\alpha \geq 0$  is a hyper-parameter that we adjust to balance the trade-off between translation quality and delay.

Similar to Gu et al. (2017), we utilize sentence-level BLEU (Papineni et al., 2002; Lin and Och, 2004) with reward shaping (Ng et al., 1999) as the reward for quality:

$$r_t^Q = \begin{cases} \Delta \text{BLEU}(\mathbf{y}^*, \mathbf{y}, t) & k \neq w_s \text{ or } s \neq S \\ \text{BLEU}(\mathbf{y}^*, \mathbf{y}) & k = w_s \text{ and } s = S \end{cases} \quad (11)$$

where

$$\begin{aligned} \Delta \text{BLEU}(\mathbf{y}^*, \mathbf{y}, t) \\ = \text{BLEU}(\mathbf{y}^*, \mathbf{y}_t) - \text{BLEU}(\mathbf{y}^*, \mathbf{y}_{t-1}) \end{aligned} \quad (12)$$

is the intermediate reward. Note that the higher the values of BLEU are, the more rewards the TN controller receives. Following Ma et al. (2018), we use average lagging (AL) as the reward for latency:

$$r_t^D = \begin{cases} 0 & k \neq w_s \text{ or } s \neq S \\ -[d(\mathbf{x}, \mathbf{y}) - d^*]_+ & k = w_s \text{ and } s = S \end{cases} \quad (13)$$

where

$$d(\mathbf{x}, \mathbf{y}) = \frac{1}{t_e} \sum_{t=1}^{t_e} l(t) - \frac{t-1}{\lambda}. \quad (14)$$

$l(t)$  is the number of observed source tokens when generating the  $t$ -th target token,  $t_e =$

Dataset	Train	Validation	Test
IWSLT16	193,591	993	1,305
WMT15	3,745,796	3,003	2,169
NIST	1,252,977	878	4,103

Table 1: # sentences in each dataset.

$\text{argmin}_t (l(t) = |\mathbf{x}|)$  denotes the earliest point when the system observes the full source sentence,  $\lambda = \frac{|\mathbf{y}|}{|\mathbf{x}|}$  represents the target-to-source length ratio and  $d^* \geq 0$  is a hyper-parameter called target delay that indicates the desired system latency. Note that the lower the values of AL are, the more rewards the TN controller receives.

**Policy Gradient** We train the TN controller with policy gradient(Sutton et al., 1999), and the gradients are:

$$\nabla_{\theta} \mathcal{J} = \mathbb{E}_{\pi_{\theta}} \left[ \sum_{t=1}^{T_{\tau}} R_t \nabla_{\theta} \log \pi_{\theta}(a_t | \cdot) \right], \quad (15)$$

where  $R_t = \sum_{i=t}^{T_{\tau}} r_i$  is the cumulative future rewards for the current decision. We can adopt any sampling approach (Chen et al., 2017, 2018; Shen et al., 2018) to estimate the expected gradient. In our experiments, we randomly sample multiple action trajectories from the current policy  $\pi_{\theta}$  and estimate the gradient with the collected accumulated reward. We try the variance reduction techniques by subtracting a baseline average reward estimated by a linear regression model from  $R_t$  and find that it does not help to improve the performance. Therefore, we just normalize the reward in each mini-batch without using baseline reward for simplicity.

## 4 Experiments

### 4.1 Settings

**Dataset** We compare our approach with the baselines on WMT15 German-English<sup>2</sup> (DE-EN) translation in both directions. This is also the most widely used dataset to evaluate SNMT’s performance (Cho and Esipova, 2016; Gu et al., 2017; Ma et al., 2018; Arivazhagan et al., 2019; Zheng et al., 2019). To further evaluate our approach’s efficacy in trading off translation quality and latency on other language pair and spoken language, we also

conduct experiments with the proposed LE and TN methods on NIST Chinese-to-English<sup>3</sup> (ZH→EN) translation and IWSLT16 German-English<sup>4</sup> (DE-EN) translation in both directions. For WMT15, we use newstest2014 for validation and newstest2015 for test. For NIST, we use MT02 for validation, and MT05, MT06, MT08 for test. For IWSLT16, we use tst13 for validation and tst14 for test. All the data is tokenized and segmented into subword symbols using byte-pair encoding (Sennrich et al., 2016) to restrict the size of the vocabulary. We use 40,000 joint merge operations on WMT15, and 24,000 on IWSLT16. For NIST, we use 30,000 merge operations for source and target side separately. Without explicitly mention, we simulate simultaneous translation scenario at inference time with these datasets by assuming that the system observes one new source token at each outer step, i.e.,  $c_s = 1$ . Table 1 shows the data statistics.

**Pretrained NMT Model** We use Transformer (Vaswani et al., 2017) trained with maximum likelihood estimation as the pretrained CNMT model and implement our method based on fairseq-py.<sup>5</sup> We follow the setting in transformer\_iwslt\_de\_en for IWSLT16 dataset, and transformer\_wmt\_en\_de for WMT15 and NIST dataset. Fairseq-py adds an EOS token for all source sentences during training and inference. Therefore, to be consistent with the CNMT model implemented with fairseq-py, we also add an EOS token at the end of the source prefix for prefix translation and find that the EOS helps translation.

**TN Controller** To train the TN controller, we use a mini-batch size of 8,16,16 and sample 5,10,10 trajectories for each sentence pair in a batch for IWSLT16, WMT15 and NIST, respectively. We set the number of newly observed source tokens at each outer step to be 1 during the training for simplicity. We set  $\alpha$  to be 0.04, and  $d^*$  to be 2, 5, 8. All our TN controllers are trained with policy gradient using Adam optimizer (Kingma and Ba, 2015) with 30,000 updates. We select the last model as our final TN controller.

**Baseline** We compare our model against three baselines that utilize a pretrained CNMT model to

<sup>3</sup>These sentence pairs are mainly extracted from LDC2002E18, LDC2003E07, LDC2003E14, Hansards portion of LDC2004T07, LDC2004T08 and LDC2005T06

<sup>4</sup><https://workshop2016.iwslt.org/>

<sup>5</sup><https://github.com/pytorch/fairseq>

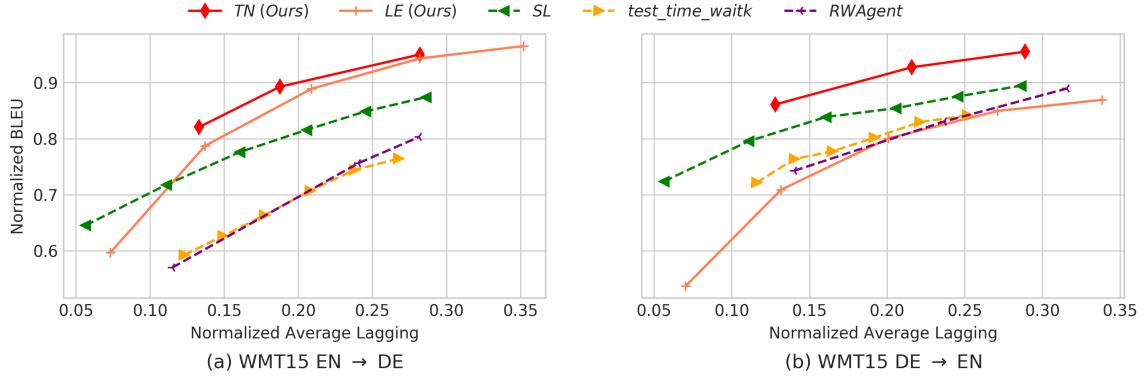


Figure 3: Comparison with the baselines on the test set of WMT15 EN→DE and WMT15 DE→EN translations. The shown points from left to right on the same line are the results of simultaneous greedy decoding with  $d^* \in \{2, 5, 8\}$  for TN,  $d \in \{0, 2, 4, 6, 8\}$  for LE,  $\rho \in \{0.65, 0.6, 0.55, 0.5, 0.45, 0.4\}$  for SL,  $k \in \{1, 3, 5, 7, 9\}$  for test\_time\_waitk and  $CW \in \{2, 5, 8\}$  for RWAgent. The scores of **Greedy** decoding: BLEU=25.16, AL=28.10 for WMT15 EN→DE translation and BLEU=26.17, AL=31.20 for WMT15 DE→EN translation.

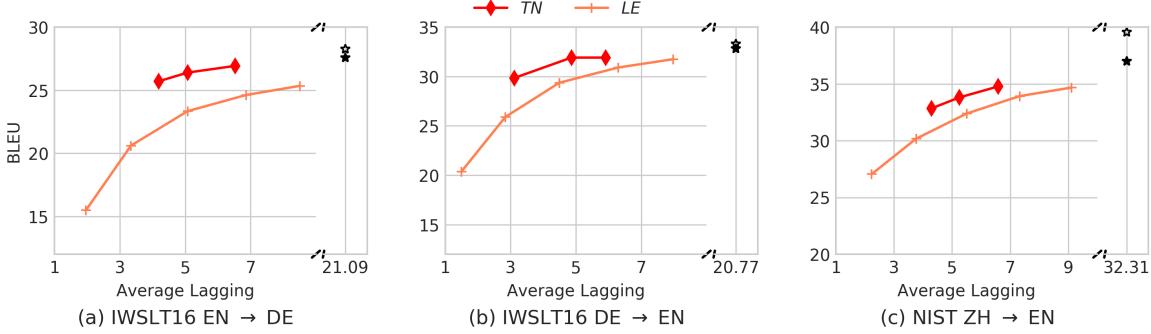


Figure 4: Performance on the test set of IWSLT16 EN→DE translation, IWSLT16 DE→EN translation and NIST ZH→EN translation. The shown points from left to right on the same line are the results of  $d^* \in \{2, 5, 8\}$  for TN and  $d \in \{0, 2, 4, 6, 7\}$  for LE. ★☆:full-sentence (greedy and beam-search).

perform simultaneous translation:

- **test\_time\_waitk** (Ma et al., 2018): the method that decodes with a waitk policy with a CNMT model. We report the results when  $k \in \{1, 3, 5, 7, 9\}$ .
- **SL** (Zheng et al., 2019): the method that adapts CNMT to SNMNT by learning an adaptive READ/WRITE policy from oracle READ/WRITE sequences generated with heuristics. We report the results with threshold  $\rho \in \{0.65, 0.6, 0.55, 0.5, 0.45, 0.4\}$ .
- **RWAgent** (Gu et al., 2017): the adaptation of Gu et al. (2017)'s full-sentence model and reinforced READ/WRITE policy network to Transformer by Ma et al. (2018). We report the results when using  $CW \in \{2, 5, 8\}$  as the target delay.

We report the result with  $d \in \{0, 2, 4, 6, 8\}$  for our proposed LE method and  $d^* \in \{2, 5, 8\}$  for our proposed TN method. For all baselines, we cite the results reported in Zheng et al. (2019).<sup>6</sup>

## 4.2 Results

We compare our methods with the baselines on the test set of WMT15 EN→DE and DE→EN translation tasks, as shown in Fig. 3. The points closer to the upper left corner indicate better overall performance, namely low latency and high quality. We observe that as latency increases, all methods improve in quality. the TN method significantly outperforms all the baselines in both translation tasks,

<sup>6</sup>Since Zheng et al. (2019) did not mention the details of data preprocessing, we cannot compare the BLEU and AL scores directly with theirs. Therefore, we normalize the BLEU and AL scores with its corresponding upper bound, i.e. the BLEU and AL scores obtained when the pretrained Transformer performs standard greedy decoding (**Greedy**).

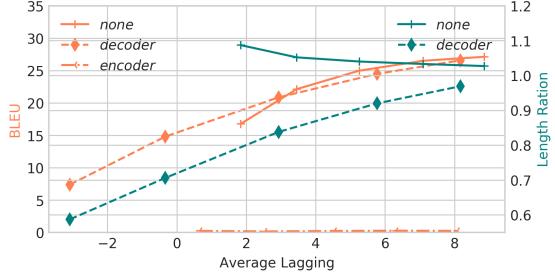


Figure 5: Comparison of whether to reuse previous encoder or decoder hidden states on WMT15 EN→DE test set with the LE controller. The left Y axis is the BLEU score and the right Y axis is the length ratio: the translation length divided by the reference length. The points on the same line are the results of  $d \in \{0, 2, 4, 6, 8\}$ . *none*: rebuild all encoder/decoder hidden states; *decoder*: reuse decoder hidden states and rebuild all encoder hidden states; *encoder*: reuse previous encoder hidden states and rebuild all decoder hidden states.

demonstrating that it indeed learns the appropriate timing to stop prefix translation. LE outperforms the baselines on WMT15 EN→DE translation at high latency region and performs similarly or worse on other cases.

We show the methods’ efficacy in trading off quality and latency on other language pair and spoken language in Fig. 4. TN outperforms LE on all translation tasks, especially at the low latency region. It obtains promising translation quality with acceptable latency: with a lag of  $< 7$  tokens, TN obtains 96.95%, 97.20% and 94.03% BLEU with respect to consecutive greedy decoding for IWSLT16 EN→DE, IWSLT16 DE→EN and NIST ZH→EN translations, respectively.

### 4.3 Analyze

We analyze the effect of different ways to obtain the encoder and decoder hidden states at the beginning of prefix translation with the LE controller. Fig. 5 shows the result. We try three variants: a) dynamically rebuild all encoder/decoder hidden states (*none*); b) reuse decoder hidden states and rebuild all encoder hidden states (*decoder*); c) reuse previous encoder hidden states and rebuild all decoder hidden states (*encoder*). The left Y axis and X axis show BLEU-vs-AL curve. We observe that if reusing previous encoder hidden states (*encoder*), the translation fails. We ascribe this to the discrepancy between training and decoding for the encoder. We also observe that when  $d \in 0, 2$ , reusing decoder hidden states (*decoder*) obtain negative AL.

To analyze this, we plot the translation to reference length ratio versus AL curve with the right Y axis and X axis. It shows that with *decoder*, the decoding process stops too early and generates too short translations. Therefore, to avoid such problem and to be consistent with the training process of the CNMT model, it is important to dynamically rebuild all encoder/decoder hidden states for prefix translation.

Since we make no assumption about the  $c_s$ , i.e., the number of newly observed source tokens at each outer step, we also test the effect of different  $c_s$ . Fig. 6 shows the result with the LE and TN controllers on the test set of WMT15 EN→DE translation. We observe that as  $c_s$  increases, both LE and TN trend to improve in quality and worsen in latency. When  $c_s = 1$ , LE controller obtains the best balance between quality and latency. In contrast, TN controller obtains similar quality and latency balance with different  $c_s$ , demonstrating that TN controller successfully learns the right timing to stop regardless of the input update schedule.

We also analyze the TN controller’s adaptability by monitoring the initial delay, i.e., the number of observed source tokens before emitting the first target token, on the test set of WMT15 EN→DE translation, as shown in Fig. 7.  $d^*$  is the target delay measured with AL (used in Eq. 13). It demonstrates that the TN controller has a lot of variance in its initial delay. The distribution of initial delay changes with different target delay: with higher target delay, the average initial delay is larger. For most sentences, the initial delay is within 1 – 7.

In speech translation, listeners are also concerned with long silences during which no translation occurs. Following Gu et al. (2017); Ma et al. (2018), we use Consecutive Wait (CW) to measure this:

$$CW(\mathbf{x}, \mathbf{y}) = \frac{\sum_{s=1}^S c_s}{\sum_{s=1}^S \mathbb{1}_{w_s > 0}}. \quad (16)$$

Fig. 8 shows the BLEU-vs-CW plots for our proposed two methods. The TN controller has higher CW than the LE controller. This is because TN controller prefers consecutive updating output buffer (e.g., it often produces  $w_s$  as 0 0 0 0 3 0 0 0 0 5 0 0 0 0 4 ...) while the LE controller often updates its output buffer following the input buffer (e.g., it often produces  $w_s$  as 0 0 0 0 1 1 1 1 1 ... when  $d = 4$ ). Although larger than LE, the CW for TN ( $< 6$ ) is acceptable for most speech translation scenarios.

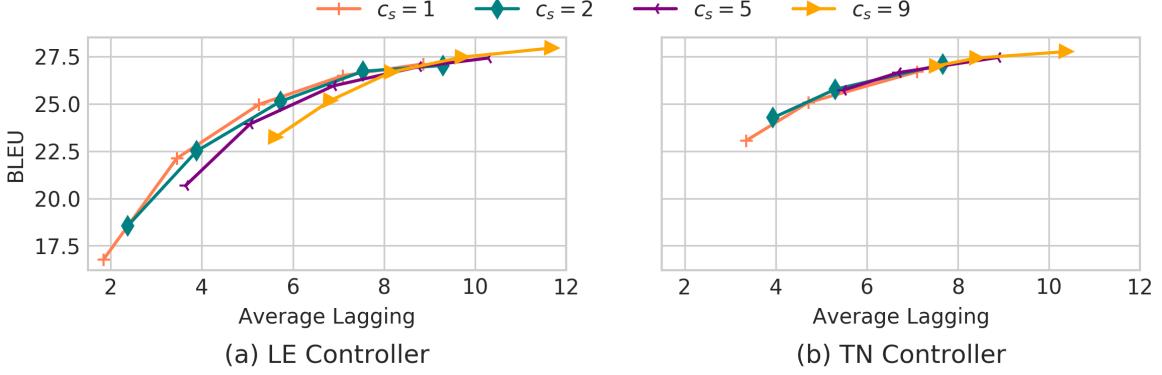


Figure 6: Performance on the test set of WMT15 EN→DE translation with different input buffer update schedule. Points on the same line are obtained by increasing  $d \in 0, 2, 4, 6, 8$  for (a) and  $d^* \in 2, 5, 8$  for (b).

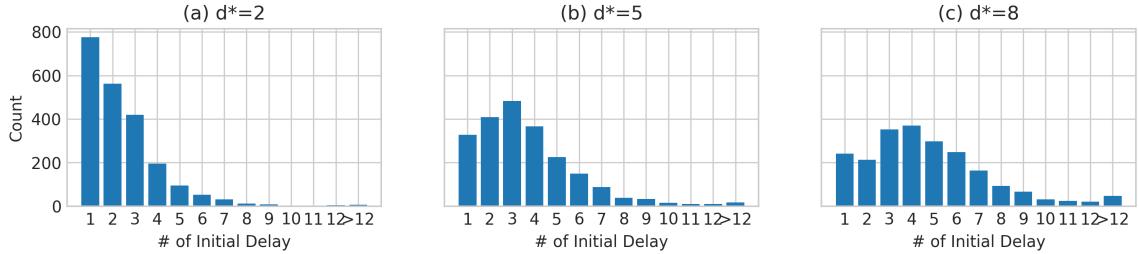


Figure 7: Number of observed source tokens before emitting the first target token for the TN controller on the test set of WMT15 EN→DE translation.

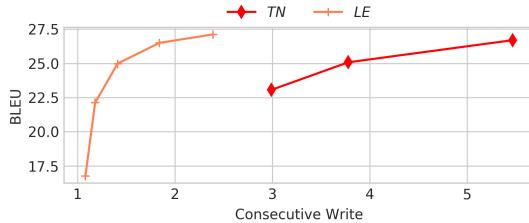


Figure 8: Average consecutive write length on the test set of WMT15 EN→DE translation.

#### 4.4 Translation Examples

Fig. 9 shows two translation examples with the LE and TN controllers on the test set of NIST ZH→EN and WMT15 EN→DE translation. In manual inspection of these examples and others, we find that the TN controller learns a conservative timing for stopping prefix translation. For example, in example 1, TN outputs translation “*wu bangguo attended the signing ceremony*” when observing “吴邦国 出席 签字 仪式 并”, instead of a more radical translation “*wu bangguo attended the signing ceremony and*”. Such strategy helps to alleviate the problem of premature translation, i.e., translating before observing enough future context.

## 5 Related Work

A number of works in simultaneous translation divide the translation process into two stages. A segmentation component first divides the incoming text into segments, and then each segment is translated by a translator independently or with previous context. The segmentation boundaries can be predicted by prosodic pauses detected in speech (Fügen et al., 2007; Bangalore et al., 2012), linguistic cues (Sridhar et al., 2013; Matusov et al., 2007), or a classifier based on alignment information (Siahbani et al., 2014; Yarmohammadi et al., 2013) and translation accuracy (Oda et al., 2014; Grissom et al., 2014; Siahbani et al., 2018).

Some authors have recently endeavored to perform simultaneous translation in the context of NMT. Niehues et al. (2018); Arivazhagan et al. (2020) adopt a re-translation approach where the source is repeatedly translated from scratch as it grows and propose methods to improve translation stability. Cho and Esipova (2016); Dalvi et al. (2018); Ma et al. (2018) introduce a manually designed criterion to control when to translate. Satija and Pineau (2016); Gu et al. (2017); Alinejad et al. (2018) extend the criterion into a trainable agent

	1	2	3	4	5	6	7	8	9
	吴邦国出席	签字仪式并				在	协议	上	签字
LE		wu				bangguo attended	the		signing ceremony and signed the agreement
TN		wu bangguo attended the signing ceremony							and signed the agreement
Greedy									wu bangguo attended the signing ceremony and signed the agreement
Ref									wu bangguo attends signing ceremony and signs agreement
	NATO does	not want to				break	agreements with	Russia	
LE		Die				NATO möchte	keine	Abkommen mit Russland brechen	
TN	Die NATO	will						keine Abkommen mit Russland brechen	
Greedy								Die NATO möchte keine Abkommen mit Russland brechen	
Ref								NATO will Vereinbarungen mit Russland nicht brechen	

Figure 9: Translation examples from the test set of NIST ZH→EN (example 1) and WMT15 EN→DE translation (example 2). We compare LE with  $d = 4$  and TN with  $d^* = 5$  because these two models achieve similar latency. Greedy and Ref represent the greedy decoding result from consecutive translation and the reference, respectively.

in a reinforcement learning framework. However, these work either develop sophisticated training frameworks explicitly designed for simultaneous translation (Ma et al., 2018) or fail to use a pre-trained consecutive NMT model in an optimal way (Cho and Esipova, 2016; Dalvi et al., 2018; Satija and Pineau, 2016; Gu et al., 2017; Alinejad et al., 2018; Zheng et al., 2019). In contrast, our work is significantly different from theirs in the way of using pretrained consecutive NMT model to perform simultaneous translation and the design of the two stopping criteria.

## 6 Conclusion

We have presented a novel framework for improving simultaneous translation with a pretrained consecutive NMT model. The basic idea is to translate partial source sentence with the consecutive NMT model and stops the translation with two novel stopping criteria. Extensive experiments demonstrate that our method with trainable stopping controller outperforms the state-of-the-art baselines in balancing between translation quality and latency.

## Acknowledgments

We thank the anonymous reviewers for their insightful feedback on this work. Yun Chen is partially supported by the Fundamental Research Funds for the Central Universities and the funds of Beijing Advanced Innovation Center for Language Resources (No. TYZ19005).

## References

- Ashkan Alinejad, Maryam Siahbani, and Anoop Sarkar. 2018. Prediction improves simultaneous neural machine translation. In *EMNLP*.
- Naveen Arivazhagan, Colin Cherry, Wolfgang Macherey, Chung-Cheng Chiu, Semih Yavuz, Ruoming Pang, Wei Li, and Colin Raffel. 2019. Monotonic infinite lookback attention for simultaneous machine translation. *ArXiv*, abs/1906.05218.
- Naveen Arivazhagan, Colin Cherry, Isabelle Te, Wolfgang Macherey, Pallavi Baljekar, and George Foster. 2020. Re-translation strategies for long form, simultaneous, spoken language translation. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7919–7923. IEEE.
- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of ICLR*.
- Srinivas Bangalore, Vivek Kumar Rangarajan Sridhar, Prakash Kolan, Ladan Golipour, and Aura Jimenez. 2012. Real-time incremental speech-to-speech translation of dialogs. In *HLT-NAACL*.
- Yun Chen, Yang Liu, Yong Cheng, and Victor OK Li. 2017. A teacher-student framework for zero-resource neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1925–1935.
- Yun Chen, Yang Liu, and Victor OK Li. 2018. Zero-resource neural machine translation with multi-agent communication game. In *AAAI*.
- Kyunghyun Cho and Masha Esipova. 2016. Can neural machine translation do simultaneous translation. *CoRR*, abs/1606.02012.
- Fahim Dalvi, Nadir Durrani, Hassan Sajjad, and Stephan Vogel. 2018. Incremental decoding and training methods for simultaneous translation in neural machine translation. In *NAACL-HLT*.
- Christian Fügen, Alexander H. Waibel, and Muntsin Kolss. 2007. Simultaneous translation of lectures and speeches. *Machine Translation*, 21:209–252.

- Alvin Grissom, He He, Jordan L. Boyd-Graber, John Morgan, and Hal Daumé. 2014. Don’t until the final verb wait: Reinforcement learning for simultaneous machine translation. In *EMNLP*.
- Jiatao Gu, Graham Neubig, Kyunghyun Cho, and Victor O. K. Li. 2017. Learning to translate in real-time with neural machine translation. In *EACL*.
- Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.
- Chin-Yew Lin and Franz Josef Och. 2004. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *ACL*.
- Mingbo Ma, Liang Huang, Hao Xiong, Kaibo Liu, Chuanqiang Zhang, Zhongjun He, Hairong Liu, Xing Li, and Haifeng Wang. 2018. Stacl: Simultaneous translation with integrated anticipation and controllable latency. *CoRR*, abs/1810.08398.
- Evgeny Matusov, Dustin Hillard, Mathew Magimai-Doss, Dilek Z. Hakkani-Tür, Mari Ostendorf, and Hermann Ney. 2007. Improving speech translation with automatic boundary prediction. In *INTERSPEECH*.
- Andrew Y. Ng, Daishi Harada, and Stuart J. Russell. 1999. Policy invariance under reward transformations: Theory and application to reward shaping. In *ICML*.
- Jan Niehues, Thai Son Nguyen, Eunah Cho, Thanh-Le Ha, Kevin Kilgour, Markus Müller, Matthias Sperber, Sebastian Stüker, and Alex Waibel. 2016. Dynamic transcription for low-latency speech translation. In *Interspeech*, pages 2513–2517.
- Jan Niehues, Ngoc-Quan Pham, Thanh-Le Ha, Matthias Sperber, and Alex Waibel. 2018. Low-latency neural speech translation. *arXiv preprint arXiv:1808.00491*.
- Yusuke Oda, Graham Neubig, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. 2014. Optimizing segmentation strategies for simultaneous speech translation. In *ACL*.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.
- Harsh Satija and Joelle Pineau. 2016. Simultaneous machine translation using deep reinforcement learning. *Abstraction in Reinforcement Learning Workshop, ICML*.
- Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.
- Shi-qi Shen, Yun Chen, Cheng Yang, Zhi-yuan Liu, Mao-song Sun, et al. 2018. Zero-shot cross-lingual neural headline generation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(12):2319–2327.
- Maryam Siahbani, Ramtin Mehdizadeh Seraj, Baskaran Sankaran, and Anoop Sarkar. 2014. Incremental translation using hierarchical phrase-based translation system. *2014 IEEE Spoken Language Technology Workshop (SLT)*, pages 71–76.
- Maryam Siahbani, Hassan Shavarani, Ashkan Alinejad, and Anoop Sarkar. 2018. Simultaneous translation using optimized segmentation. In *AMTA*.
- Vivek Kumar Rangarajan Sridhar, John Chen, Srinivas Bangalore, Andrej Ljolje, and Rathinavelu Chengalvarayan. 2013. Segmentation strategies for streaming speech translation. In *HLT-NAACL*.
- Richard S. Sutton, David A. McAllester, Satinder P. Singh, and Yishay Mansour. 1999. Policy gradient methods for reinforcement learning with function approximation. In *NIPS*.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.
- Mahsa Yarmohammadi, Vivek Kumar Rangarajan Sridhar, Srinivas Bangalore, and Baskaran Sankaran. 2013. Incremental segmentation and decoding strategies for simultaneous translation. In *IJCNLP*.
- Baigong Zheng, Renjie Zheng, Mingbo Ma, and Liang Huang. 2019. Simpler and faster learning of adaptive policies for simultaneous translation. In *EMNLP*, pages 1349–1354, Hong Kong, China. Association for Computational Linguistics.

# UnihanLM: Coarse-to-Fine Chinese-Japanese Language Model Pretraining with the Unihan Database

Canwen Xu<sup>1\*</sup>, Tao Ge<sup>2</sup>, Chenliang Li<sup>3</sup>, Furu Wei<sup>2</sup>

<sup>1</sup> University of California, San Diego <sup>2</sup> Microsoft Research Asia <sup>3</sup> Wuhan University

<sup>1</sup> cxu@ucsd.edu <sup>2</sup> {tage, fuwei}@microsoft.com <sup>3</sup> cllee@whu.edu.cn

## Abstract

Chinese and Japanese share many characters with similar surface morphology. To better utilize the shared knowledge across the languages, we propose UnihanLM, a self-supervised Chinese-Japanese pretrained masked language model (MLM) with a novel two-stage coarse-to-fine training approach. We exploit Unihan, a ready-made database constructed by linguistic experts to first merge morphologically similar characters into clusters. The resulting clusters are used to replace the original characters in sentences for the coarse-grained pretraining of the MLM. Then, we restore the clusters back to the original characters in sentences for the fine-grained pretraining to learn the representation of the specific characters. We conduct extensive experiments on a variety of Chinese and Japanese NLP benchmarks, showing that our proposed UnihanLM is effective on both mono- and cross-lingual Chinese and Japanese tasks, shedding light on a new path to exploit the homology of languages.<sup>1</sup>

## 1 Introduction

Recently, Pretrained Language Models have shown promising performance on many NLP tasks (Peters et al., 2018; Devlin et al., 2019; Liu et al., 2019; Yang et al., 2019c; Lan et al., 2020). Many attempts have been made to train a model that supports multiple languages. Among them, Multilingual BERT (mBERT) (Devlin et al., 2019) is released as a part of BERT. It directly adopts the same model architecture and training objective, and is trained on Wikipedia in different languages. XLM (Lample and Conneau, 2019) is proposed with an additional language embedding and a new training

\* This work was done during Canwen’s internship at Microsoft Research Asia.

<sup>1</sup>The code and pretrained weights are available at <https://github.com/JetRunner/unihan-lm>.

JA	台 <sup>1</sup> 風 <sup>2</sup> は熱 <sup>3</sup> 帶 <sup>4</sup> 低氣 <sup>5</sup> 壓 <sup>6</sup> の <u>一</u> 種 <sup>7</sup> です。
T-ZH	颱 <sup>1</sup> 風 <sup>2</sup> 是 <u>熱</u> <sup>3</sup> <u>帶</u> <sup>4</sup> <u>氣</u> <sup>5</sup> <u>壓</u> <sup>6</sup> 的 <u>一</u> 種 <sup>7</sup> 。
S-ZH	台 <sup>1</sup> 风 <sup>2</sup> 是热 <sup>3</sup> 带 <sup>4</sup> 低气 <sup>5</sup> 压 <sup>6</sup> 的 <u>一</u> 种 <sup>7</sup> 。
EN	Typhoon is a type of tropical depression.

Table 1: A sentence example in Japanese (JA), Traditional Chinese (T-ZH) and Simplified Chinese (S-ZH) with its English translation (EN). The characters that already share the same Unicode are marked with an underline. In this work, we further match characters with identical meanings but different Unicode, then merge them. Characters eligible to be merged together are marked with the same superscript.

objective (translation language modeling, TLM). XLM-R (Conneau et al., 2019) has a larger size and is trained with more data. Based on XLM, UniCoder (Huang et al., 2019) collects more data and uses multi-task learning to train on three supervised tasks.

The census of cross-lingual approaches is to allow lexical information to be shared between languages. XLM and mBERT exploit shared lexical information by Byte Pair Encoding (BPE) (Sennrich et al., 2016) and WordPiece (Wu et al., 2016), respectively. However, these automatically learned shared representations have been criticized by recent work (K et al., 2020), which reveals their limitations in sharing meaningful semantics across languages. Also, words in both Chinese and Japanese are short, which prohibits an effective learning of sub-word representations. Different from European languages, Chinese and Japanese naturally share Chinese characters as a subword component. Early work (Chu et al., 2013) shows that shared characters in these two languages can benefit Example-based Machine Translation (EBMT) with a statistical based phrase extraction and alignment. For Neural Machine Translation (NMT), (Zhang and Ko-

machi, 2019) exploited such information by learning a BPE representation over sub-character (i.e., ideograph and stroke) sequence. They applied this technique to unsupervised Chinese-Japanese machine translation and achieved state-of-the-art performance. However, this approach greatly relies on unreliable automatic BPE learning and may suffer from the noise brought by various variants.

To facilitate lexical sharing, we propose **Unihan Language Model** (UnihanLM), a cross-lingual pre-trained masked language model for Chinese and Japanese. We propose a two-stage coarse-to-fine pretraining procedure to empower better generalization and take advantages of shared characters in Japanese, Traditional and Simplified Chinese. First, we let the model exploit maximum possible shared lexical information. Instead of learning a shared sub-word vocabulary like the prior work, we leverage Unihan database (Jenkins et al., 2019), a ready-made constituent of the Unicode standard, to extract the shared lexical information across the languages. By exploiting this database, we can effectively merge characters with the similar surface morphology but independent Unicodes, as shown in Table 1 into thousands of clusters. The clusters will be used to replace the characters in sentences during the first-stage coarse-grained pretraining. After the coarse-grained pretraining finishes, we restore the clusters back to the original characters and initialize their representation with their corresponding cluster’s representation and then learn their specific representation during the second-stage fine-grained pretraining. In this way, our model can make full use of shared characters while maintaining a good sense for nuances of similar characters.

To verify the effectiveness of our approach, we evaluate on both lexical and semantic tasks in Chinese and Japanese. On word segmentation, our model outperforms monolingual and multi-lingual BERT (Devlin et al., 2019) and shows a much higher performance on cross-lingual zero-shot transfer. Also, our model achieves state-of-the-art performance on unsupervised Chinese-Japanese machine translation, and is even comparable to the supervised baseline on Chinese-to-Japanese translation. On classification tasks, our model achieves a comparable performance with monolingual BERT and other cross-lingual models trained with the same scale of data.

To summarize, our contributions are three-fold: (1) We propose UnihanLM, a cross-lingual pre-

trained language model for Chinese and Japanese NLP tasks. (2) We pioneer to apply the language resource – the Unihan Database to help model pre-training, allowing more lexical information to be shared between the two languages. (3) We devise a novel coarse-to-fine two-stage pretraining strategy with different granularity for Chinese-Japanese language modeling.

## 2 Preliminaries

### 2.1 Chinese Character

Chinese character is a pictograph used in Chinese and Japanese. These characters often share the same background and origin. However, due to historic reasons, Chinese characters have developed into different writing systems, including Japanese Kanji, Traditional Chinese and Simplified Chinese. Also, even in a single text, multiple variants of the same characters can be used interchangeably (e.g., “台灣” and “臺灣” for “Taiwan”, in Traditional Chinese). These characters have identical or overlapping meanings. Thus, it is critical to better exploit such information for modeling both cross-lingual (i.e., between Chinese and Japanese), cross-system (i.e., between Traditional and Simplified Chinese) and cross-variant semantics.

Both Chinese and Japanese have no delimiter (e.g., white space) to mark the boundaries of words. There have always been debates over whether word segmentation is necessary for Chinese NLP. Recent work (Li et al., 2019) concludes that it is not necessary for various NLP tasks in Chinese. Previous cross-lingual language models use different methods for tokenization. mBERT adds white spaces around Chinese characters and lefts Katakana/Hiragana Japanese (also known as kanas) unprocessed. Different from mBERT, XLM uses Stanford Tokenizer<sup>2</sup> and KyTea<sup>3</sup> to segment Chinese and Japanese sentences, respectively. After tokenization, mBERT and XLM use WordPiece (Wu et al., 2016) and Byte Pair Encoding (Sennrich et al., 2016) for sub-word encoding, respectively.

Nevertheless, both approaches suffer from obvious drawbacks. For mBERT, the kanas and Chinese characters are treated differently, which causes a mismatch for labeling tasks. Also, leaving kanas untokenized may cause the data sparsity problem. For XLM, as pointed out in (Li et al., 2019), an

<sup>2</sup><https://nlp.stanford.edu/software/tokenizer.html>

<sup>3</sup><http://www.phontron.com/kytea/>

Variant	Description	Example
Traditional Variant	The traditional versions of a simplified Chinese character.	发 → 髮 (hair), 發 (to burgeon)
Simplified Variant	The simplified version of a traditional Chinese character.	團 → 团 (group)
Z-Variant	Same character with different unicodes only for compatibility.	説 ↔ 說 (say)
Semantic Variant	Characters with identical meaning.	兎 ↔ 兔 (rabbit)
Specialized Semantic Variant	Characters with overlapping meaning.	丂 (rice bowl, well) ↔ 丂 (well)

Table 2: The five types of variants in the Unihan database.

external word segmenter would introduce extra segmentation errors and compromise the performance of the model. Also, as a word-based model, it is difficult to share cross-lingual characters unless the segmented words in both Chinese and Japanese are exactly matched. Furthermore, both approaches would enlarge the vocabulary size and thus introduce more parameters.

## 2.2 Unihan Database

Chinese, Japanese and Korean (CJK) characters share a common origin from the ancient Chinese characters. However, with the development of each language, both the shape and semantics of characters drastically change. When exchanging information, different codings of the same character hinders the text processing. Thus, as the result of Han unification<sup>4</sup>, the database of CJK Unified Ideographs, Unihan (Jenkins et al., 2019), is constructed by human experts tracing the sources of each character.

As part of the Unicode Standard, Unihan merges the Unicode for some characters from different languages and provides extra variant information between different characters. In previous studies (Zhang and Komachi, 2019; Lample and Conneau, 2019; Devlin et al., 2019), Unicode is used by default. However, due to the “Source Separation Rule” of Unicode, to remain the compatibility with prior encoding systems, a single character can have multiple Unicodes with different glyphs. For example, for the character “戶”, there are three unicodes: U+6236, U+6237 and U+6238. This feature could be useful for message exchange but is undoubtedly undesirable for NLP and may bring the problems of data sparsity and prevent the alignment of a cross-lingual language model.

Fortunately, Unihan database also provides 12,373 entries of variant information in five types, as listed in Table 2. Note that one character may have multiple types of variants and each type may

Tokenization Scheme	Result
BERT (2019)	台風 / はひどい
XLM (2019)	台風 / は / ひどい
UnihanLM	台 / 風 / は / ひ / ど / い

Table 3: Different tokenization schemes used in recent work and ours. Note that the tokenized results of both BERT and XLM in this table are before Word-Piece/BPE applied. WordPiece/BPE may further split a token.

have multiple variant characters (e.g., the traditional variants of “发” in Table 2). Such information forms a complex graph structure.

## 3 UnihanLM

In this section, we introduce the tokenization, character merging and training procedure for our proposed UnihanLM.

### 3.1 Tokenization

As analyzed in Section 2.1, the tokenization scheme is tricky and critical for East Asian languages. Although recent work (Li et al., 2019) reveals that tokenization is unnecessary for most high-level NLU and NLG tasks, many downstream labeling tasks (e.g., Part-of-speech Tagging, Named Entity Recognition) still require an implicit or explicit segmentation. To enable all NLP tasks, we tokenize the sentences by treating every character (including Japanese Kana) as a token. Thus, our model is capable of processing all tasks, from the lowest-level Chinese and Japanese word segmentation to high-level NLU tasks. We summarize the different tokenization schemes used in recent work and ours in Table 3.

We do not further apply BPE to our tokenized sentences for two reasons. First, a character is the atomic element in both Chinese and Japanese grammars which should not be further split. Second, character itself is naturally a sub-word semantic element, e.g., “自” (self) + “信”(belief) = “自信”

<sup>4</sup>[https://en.wikipedia.org/wiki/Han\\_unification](https://en.wikipedia.org/wiki/Han_unification)

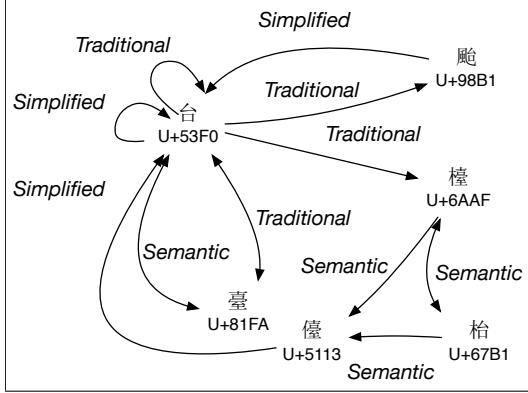


Figure 1: A connected subgraph of Unihan database. For example, for the word “typhoon”, “台” is used in Japanese and Simplified Chinese while “颱” is used in Traditional Chinese.

(confidence); “自” (self) + “尊”(respect) = “自尊” (self-esteem).

### 3.2 Character Merging

To reduce the vocabulary size and align the Chinese characters in Traditional Chinese, Simplified Chinese and Japanese to the greatest extent, it is important to merge as many characters as possible while ensuring only merging characters with the identical or overlapping meanings. Thus, we use Unihan database, which includes character variant information collected and approved by human experts. We use four types of variants including Traditional Variant, Simplified Variant, Z-Variant and Semantic Variant. Note that we exclude Specialized Semantic Variant which may raise ambiguity problem since it is not very common and the semantics of the two characters are merely overlapping, not identical.

However, merging characters is still challenging since the variant information in Unihan database is a complex graph, as illustrated in Figure 1. To merge the characters as much as possible, we convert Unihan database to a large undirected graph and use Union Find Algorithm (Galler and Fischer, 1964) to find all maximal connected subgraph. For example, the whole Figure 1 is a subgraph in the Unihan graph found by the algorithm. We call all characters in a maximal connected subgraph belong to a “cluster”. After this merging procedure, the 12,373 variant entries yield a total of 4,001 clusters.

### 3.3 Training Procedure

As illustrated in Figure 2, the model is a Transformer based model with three embeddings as in-

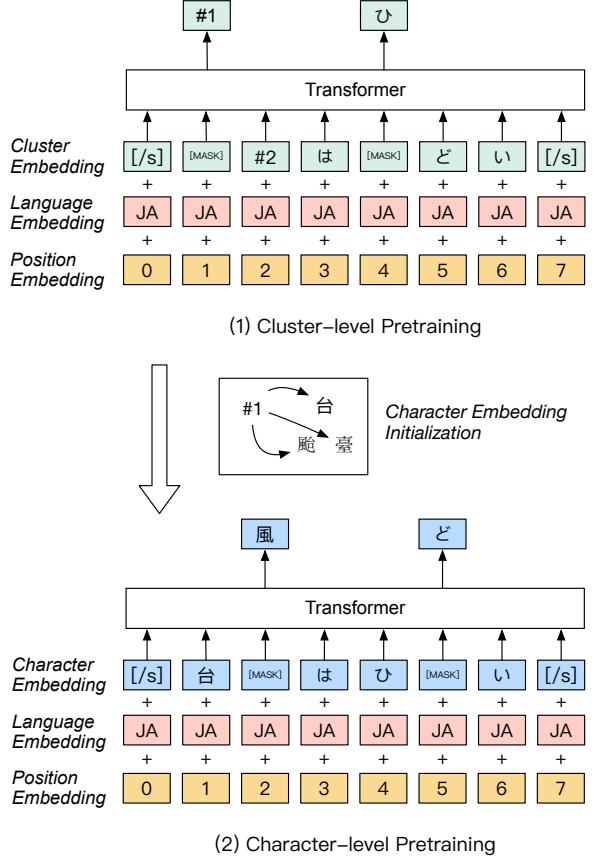


Figure 2: The model architecture of UnihanLM. (1) We merge characters to clusters and use cluster indices when doing cluster-level pretraining. In the figure, “#1” and “#2” indicate indices of the clusters which “台” and “風” belong to, respectively. (2) We initialize the embedding of each character in a cluster with the cluster embedding and do character-level pretraining to predict each character.

put and the training procedure is composed of two phases.

#### 3.3.1 Model

Our model is a Transformer-based Masked Language Model (Devlin et al., 2019) which learns to predict the randomly masked words with the context. Also, following (Lample and Conneau, 2019), we add language embedding to help the model distinguish between Chinese and Japanese, especially when we share the characters between these two languages. The detailed hyperparameter settings are described in Section 4.1.

#### 3.3.2 Coarse-grained Cluster-level Pretraining

To maximize the shared lexicon and force them to share a representation, we leverage clusters to pre-train our models on a coarse-grained cluster level.

We first append the cluster indices to the vocabulary. During cluster-level pretraining, we substitute the character index with its corresponding cluster index if the character is in the Unihan database. For Japanese kanas, punctuation, number and other characters not in Unihan database, we keep its original token index. In this way, we employ human prior knowledge to the pretraining procedure and allow the model to roughly model the semantic knowledge.

### 3.3.3 Fine-grained Character-level Pretraining

Although the clusters training is effective, there are two problems remaining unsolved. First, Traditional Variant could be ambiguous. As shown in Table 2, a character (most likely one used in Simplified Chinese) may have multiple Traditional Variants. Although it should not have a significant negative effect for understanding the language (since a Simplified Chinese user can disambiguate between different meanings of a character based on its context), it still makes sense to improve the overall performance by distinguish the characters explicitly (Navigli et al., 2017). Also, in tasks involving decoding (e.g., machine translation), they must be processed independently. Thus, character disambiguation can be naturally used as a self-supervised task. Second, when using the trained model for translation, it would be important for the model to decode the right character for different languages and writing systems. For example, for the word meaning “typhoon”, “台風”, “颱風”, “台风” should be used in Japanese, Traditional Chinese and Simplified Chinese, respectively.

Consequently, we leave these nuances of characters to a fine-grained character-level pretraining. Since during the cluster-level pretraining, all characters in Unihan database are preserved in the vocabulary but their embedding is untrained, we initialize their embedding with their corresponding cluster embedding trained in cluster-level pretraining stage. In the character-level pretraining stage, we discard the clusters in the vocabulary and do not substitute any character since then. In this way, the model can handle each character case by case, with a fine granularity. We restart the training with a smaller learning rate to allow the model to learn to disambiguate.

Model	#Layer	#Param.
BERT-Mono-ZH (2019)	12	110M
mBERT (2019)	12	179M
XLM (2019)	16	571M
UnihanLM	12	176M

Table 4: The numbers of layers and parameters for different models.

## 4 Experiments

In this section, we compare UnihanLM with other self-supervised pretrained language models. All of our baselines (monolingual BERT, mBERT and XLM) use Wikipedia for self-supervised pre-training. Note that we do not compare our model to XLM-R (Conneau et al., 2019) and UniCoder (Huang et al., 2019) since they are trained with much more data and even on supervised tasks.

### 4.1 Training Details

We use the mixture of Chinese and Japanese Wikipedia<sup>5</sup> as the unparalleled pretraining corpus. We sample 5,000 sentences as validation set for model selection and use the rest for training. Our model uses 12 layers of Transformer blocks with 16 attention heads. The hidden size is set to 1,024. The vocabulary size is 24,044. Shown in Table 4, our model has a similar size to mBERT. We train our model on 8 Nvidia V100 32GB GPUs to optimize Masked Language Model (MLM) objective (Devlin et al., 2019) with an Adam (Kingma and Ba, 2015) optimizer. The masking probability is set to 15%. We add a L2 regularization of 0.01. We warm up the first 30,000 steps for each stage of pretraining by an inverse square root function. The batch size is set to 64 per GPU. The maximum sequence length is limited to 256 tokens. We add dropout (Srivastava et al., 2014) for both feed-forward network and attention with a drop rate of 0.1. The learning rate for cluster-level pretraining is set to  $1 \times 10^{-4}$ . After 264 hours of cluster-level pretraining until convergence, we perform character-level pretraining with a smaller learning rate of  $5 \times 10^{-5}$  for another 43 hours. We choose the best model according to its perplexity on validation set. For downstream tasks (to be detailed shortly), we fine-tune UnihanLM with a learning rate of  $5 \times 10^{-7}$ ,  $1 \times 10^{-4}$ ,  $2.5 \times 10^{-5}$  and a batch

<sup>5</sup><https://dumps.wikimedia.org/>

Method	PKU (ZH)	KWDLC (JA)
<i>Standard training</i>		
mBERT (2019)	95.0	96.3
BERT-Mono-ZH (2019)	96.5	-
UnihanLM	<b>96.6</b>	<b>98.2</b>
<i>Cross-lingual zero-shot transfer</i>		
mBERT (2019)	82.0	63.1
UnihanLM	<b>85.7</b>	<b>74.1</b>

Table 5: F1 scores on Chinese Word Segmentation (CWS) and Japanese Word Segmentation (JWS) tasks. “Cross-lingual zero-shot transfer” indicates that the model is trained on CWS and zero-shot tested on JWS, vice versa.

size of 20, 24, 16 for word segmentation, unsupervised machine translation and classification tasks, respectively.

## 4.2 Word Segmentation

Word segmentation is a fundamental task in both Chinese and Japanese NLP. It is often recognized as the first step for further processing in many systems. Thus, we evaluate Chinese Word Segmentation (CWS) and Japanese Word Segmentation (JWS) on PKU dataset (Emerson, 2005) and KWDLC (Kawahara et al., 2014). We use Multilingual BERT and monolingual Chinese BERT (Devlin et al., 2019) as baselines. We use pretrained checkpoints provided by Google<sup>6</sup>. Following previous work, we treat the word segmentation task as a sequence labeling task. Note that XLM (Lample and Conneau, 2019) uses pre-segmented sentences as input, making it inapplicable for this task. As shown in Table 5, our proposed UnihanLM outperforms mBERT and monolingual BERT by 1.6 and 0.1 in terms of F1 score on CWS, respectively. On JWS, our model outperforms mBERT by 1.9 on F1. Additionally, we conduct zero-shot transfer experiments to determine how much lexical knowledge is shared within Chinese and Japanese for each model. We use the weights trained on CWS and JWS for zero-shot transferring on the other language. Our model drastically outperforms mBERT on this task by 3.7 and 11.0 on CWS and JWS, respectively. This proves that our model can better capture the lexical knowledge shared between Chinese and Japanese. Also, it is notable that zero-shot JWS has a prominently poorer performance than zero-shot CWS. As we

Method	ZH→JA	JA→ZH
<i>Supervised baseline</i>		
OpenNMT (Klein et al., 2017)	42.12	40.63
<i>Fine-tuned on Wikipedia</i>		
XLM (Lample and Conneau, 2019)	14.58	15.06
UnihanLM	<b>33.53</b>	<b>28.70</b>
<i>Fine-tuned on shuffled ASPEC-JC training set</i>		
Stroke (Zhang and Komachi, 2019)	33.81	31.66
UnihanLM	<b>44.59</b>	<b>40.58</b>

Table 6: BLEU scores of Chinese-Japanese unsupervised translation on ASPEC-JC dataset.

analyze, the criterion for segmenting Chinese characters can be learned with a Japanese corpus and then transferred to CWS. However, since no kana is present in CWS, the model cannot successfully segment kanas, when performing zero-shot inference on JWS.

## 4.3 Unsupervised Machine Translation

A Chinese speaker who never learned Japanese can roughly understand a Japanese text (and vice versa), due to the similarity between the writing systems of these two languages. On the other hand, only a few parallel corpora between Chinese and Japanese are publicly available, and they are usually small in size. Thus, Unsupervised Machine Translation (UMT) is very promising and meaningful on the Chinese-Japanese translation task. We evaluate on Asian Scientific Paper Excerpt Corpus Japanese-Chinese (ASPEC-JC)<sup>7</sup>, the most widely-used Chinese-Japanese Machine Translation dataset. We perform our experiments under two settings: (1) Chinese and Japanese Wikipedia is used as the monolingual corpora, following the setting of (Lample and Conneau, 2019). (2) Shuffled unparalleled ASPEC-JC training set is used as the monolingual corpora, following the settings in (Zhang and Komachi, 2019).

Except for XLM, we choose (Zhang and Komachi, 2019), the current state-of-the-art Chinese-Japanese UMT model as a strong baseline. They decomposed a Chinese character in both Chinese and Japanese into strokes and then learn a shared token in the stroke sequence to increase the shared tokens in the vocabulary. However, this method relies on an unsupervised BPE (Sennrich et al., 2016) to learn shared stroke tokens from a long noisy

<sup>6</sup><https://github.com/google-research/bert>

<sup>7</sup><http://orchid.kuee.kyoto-u.ac.jp/ASPEC/>

Method	PAWS-X	
	ZH	JA
BOW	54.5	55.1
ESIM (Chen et al., 2017a)	60.3	59.6
mBERT (Devlin et al., 2019)	82.3	79.2
XLM (Lample and Conneau, 2019)	82.5	79.5
UnihanLM	<b>82.7</b>	<b>80.5</b>

Table 7: Accuracy scores on PAWS-X dataset.

stroke sequence, which is rather unreliable compared to our solution. For example, “丑” (ugly) and “五” (five) have a very similar stroke sequence but completely different meanings. Following (Lample and Conneau, 2019), we use our pretrained weights to initialize the translation model and train the model with denoising auto-encoding loss and online back-translation loss. Note that both baselines use Wikipedia as the unsupervised data and are based on the same UMT method (Lample et al., 2018c). We use character-level BLEU (Papineni et al., 2002) as the evaluation metric.

We demonstrate the results in Table 6. As we analyzed, XLM suffers from a severe out-of-vocabulary (OOV) problem on AESPEC-JC, a dataset composed of scientific papers, containing many new terminologies which do not show up in the pretraining corpus of XLM. As a word-based model, XLM is not able to handle these new words and thus yields a rather poor result. When finetuned on unparalleled training set of ASPEC-JC, our model outperforms the previous state-of-the-art model (Zhang and Komachi, 2019) by a large margin of 10.78 and 8.92 in terms of BLEU. Also notably, UnihanLM even outperforms the supervised baseline on Chinese-to-Japanese translation and has a performance in close proximity on Japanese-to-Chinese task, compared to an early supervised machine translation model, OpenNMT (Klein et al., 2017), trained on the paired training set of ASPEC-JC.

#### 4.4 Text Classification

To further evaluate our model, we perform our experiments on Cross-lingual Paraphrase Aversaries from Word Scrambling (PAWS-X) (Yang et al., 2019b), a newly proposed cross-lingual text classification dataset supporting seven languages including Chinese and Japanese. This dataset consists of challenging English paraphrase identification pairs from Wikipedia and Quora. Then the human trans-

lators translate the text into the other six languages. We test under the setting of *TRANSLATE-TRAIN* (i.e., we use the provided translation of the training set for both Chinese and Japanese and test in the same language). Shown in Table 7, UnihanLM outperforms all baselines in (Yang et al., 2019b), including mBERT.

#### 4.5 Ablation Study

To verify the effectiveness of our two-stage pre-training procedure, we conduct an ablation study. A character-level model is trained from scratch without the cluster-level pretraining and marked as “*–cluster*”. On the other hand, we use the model trained in cluster-level stage for downstream tasks and mark it as “*–character*”. Note that since the objective for cluster-level stage is to predict the masked cluster, it cannot be used for unsupervised translation. Shown in Figure 8, both cluster-level and character-level pretraining play an essential role on classification tasks. On translation task, cluster-level pretraining is more important when fine-tuned on Wikipedia but has a relatively smaller impact when using shuffled ASPEC-JC training set.

To analyze the success of our two-stage training strategy, we would like to emphasize two strengths. First, as mentioned before, our easy-to-hard training procedure matches the core idea of Curriculum Learning (Bengio et al., 2009), which smooths the training and help the model generalize better. Second, the two-stage procedure inherently introduces a new self-supervised task, which could take the advantage of Multitask Learning (Caruana, 1993).

### 5 Related Work

**Multilingual Representation Learning** Learning cross-lingual representations are useful for downstream tasks such as cross-lingual classification (Conneau et al., 2018; Yang et al., 2019b), cross-lingual retrieval (Zweigenbaum et al., 2017; Artetxe and Schwenk, 2019) and cross-lingual QA (Artetxe et al., 2019; Lewis et al., 2019; Clark et al., 2020). Earlier work on multilingual representations exploiting parallel corpora (Luong et al., 2015; Gouws et al., 2015) or a bilingual dictionary to learn a linear mapping (Mikolov et al., 2013; Faruqui and Dyer, 2014). Subsequent methods explored self-training (Artetxe et al., 2017) and unsupervised learning (Zhang et al., 2017; Artetxe et al., 2018; Lample et al., 2018b). Recently, multilingual pretrained encoders have shown its effec-

Method	PAWS-X		ASPEC-JC			
			Wiki		Shuffled-train	
	ZH	JA	ZH→JA	JA→ZH	ZH→JA	JA→ZH
UnihanLM	<b>82.7</b>	<b>80.5</b>	<b>33.53</b>	<b>28.70</b>	<b>44.59</b>	<b>40.58</b>
-cluster	81.5	79.2	29.33	20.93	42.34	39.24
-character	82.0	80.1	-	-	-	-

Table 8: The results of ablation study on text classification and UMT. “-cluster” and “-character” indicate the model trained without the cluster-level pretraining and character-level pretraining, respectively. The metrics for PAWS-X and ASPEC-JC are accuracy and BLEU, respectively.

tiveness for learning deep cross-lingual representations (Eriguchi et al., 2018; Pires et al., 2019; Wu and Dredze, 2019; Lample and Conneau, 2019; Conneau et al., 2019; Huang et al., 2019).

**Word Segmentation** Word segmentation is often formalized as a sequence tagging task. It requires lexical knowledge to split a character sequence into a word list that can be used for downstream tasks. This step is necessary for many earlier NLP systems for Chinese and Japanese. Recent work on Chinese Word Segmentation (Wang and Xu, 2017; Zhou et al., 2017; Yang et al., 2017; Cai et al., 2017; Chen et al., 2017b; Yang et al., 2019a) and Japanese Word Segmentation (Kaji and Kitamura, 2014; Fujinuma and II, 2017; Kitagawa and Komachi, 2018) exploit deep neural networks and focus on building end-to-end sequence tagging models.

**Unsupervised Machine Translation** Recently, machine translation systems have demonstrated near human-level performance on some languages. However, it depends on the availability of large amounts of parallel sentences. Unsupervised Machine Translation addresses this problem by exploiting monolingual corpora which can be easily constructed. Lample et al. (2018a) proposed a UMT model by learning to reconstruct in both languages from a shared feature space. Lample et al. (2018c) exploited language modeling and back-translation and thus proposed a neural unsupervised translation model and a phase-based translation model. Different from European languages (e.g., English), Chinese and Japanese naturally share Chinese characters. Zhang and Komachi (2019) exploited such information by learning a BPE representation over sub-character (i.e., ideograph and stroke) sequence. They applied this technique to unsupervised Chinese-Japanese machine translation and achieved state-of-the-art performance. This information is also shown to be

effective by (Xu et al., 2019).

## 6 Discussion and Future Work

There is still space to improve for our method. First, as we analyze, except for Chinese characters, English words often appear in both Chinese and Japanese texts. In our current model, they are treated as normal characters without any special processing. However, such a rough processing may harm the performance of the model on some tasks. For example, in PAWS-X, many entities remain untranslated and this may have a negative effect on the performance of our model. Also, loan words (i.e., Gairaigo), especially from English, constitute a large part of nouns in modern Japanese (Miller, 1998). These words are written with kanas, instead of Chinese characters which makes it inapplicable to be shared with our approach. Thus, it may be reasonable to involve English in cross-lingual modeling of Asian languages, as well. Similarly, Chinese characters exist in Korean and Vietnamese but are now written in Hangul (Korean alphabet) and Vietnamese alphabet, respectively. Our future work will explore the possibility to generalize the idea to more Asian languages including Korean and Vietnamese.

## 7 Conclusion

In this paper, we exploit the ready-made Unihan database constructed by linguistic experts and propose a novel Chinese-Japanese cross-lingual language model trained by a two-stage coarse-to-fine procedure. Our extensive experiments on word segmentation, unsupervised machine translation and text classification verify the effectiveness of our model. Our approach sheds some light on the linguistic features that receive insufficient attention recently and showcases a novel way to fuse human linguistic knowledge and exploit the similarity between two languages.

## Acknowledgments

We are grateful for the insightful comments from the anonymous reviewers. We would like to thank Longtu Zhang and Mamoru Komachi from Tokyo Metropolitan University for their help with the MT baseline.

## References

- Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2017. Learning bilingual word embeddings with (almost) no bilingual data. In *ACL*.
- Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *ACL*.
- Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2019. On the cross-lingual transferability of monolingual representations. *CoRR*, abs/1910.11856.
- Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Trans. Assoc. Comput. Linguistics*, 7:597–610.
- Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *ICML*.
- Deng Cai, Hai Zhao, Zhisong Zhang, Yuan Xin, Yongjian Wu, and Feiyue Huang. 2017. Fast and accurate neural word segmentation for chinese. In *ACL*.
- Rich Caruana. 1993. Multitask learning: A knowledge-based source of inductive bias. In *ICML*.
- Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017a. Enhanced LSTM for natural language inference. In *ACL*.
- Xinchi Chen, Zhan Shi, Xipeng Qiu, and Xuanjing Huang. 2017b. Adversarial multi-criteria learning for chinese word segmentation. In *ACL*.
- Chenhui Chu, Toshiaki Nakazawa, Daisuke Kawahara, and Sadao Kurohashi. 2013. Chinese-japanese machine translation exploiting chinese characters. *ACM Trans. Asian Lang. Inf. Process.*, 12(4):16:1–16:25.
- Jonathan H. Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. 2020. Tydi QA: A benchmark for information-seeking question answering in typologically diverse languages. *CoRR*, abs/2003.05002.
- Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116.
- Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: evaluating cross-lingual sentence representations. In *EMNLP*.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.
- Thomas Emerson. 2005. The second international chinese word segmentation bakeoff. In *SIGHAN@IJCNLP*.
- Akiko Eriguchi, Melvin Johnson, Orhan Firat, Hideto Kazawa, and Wolfgang Macherey. 2018. Zero-shot cross-lingual classification using multilingual neural machine translation. *CoRR*, abs/1809.04686.
- Manaal Faruqui and Chris Dyer. 2014. Improving vector space word representations using multilingual correlation. In *EACL*.
- Yoshinari Fujinuma and Alvin Grissom II. 2017. Substring frequency features for segmentation of japanese katakana words with unlabeled corpora. In *IJCNP*.
- Bernard A. Galler and Michael J. Fischer. 1964. An improved equivalence algorithm. *Commun. ACM*, 7(5):301–303.
- Stephan Gouws, Yoshua Bengio, and Greg Corrado. 2015. Bilbowa: Fast bilingual distributed representations without word alignments. In *ICML*.
- Haoyang Huang, Yaobo Liang, Nan Duan, Ming Gong, Linjun Shou, Dixin Jiang, and Ming Zhou. 2019. Unicoder: A universal language encoder by pre-training with multiple cross-lingual tasks. In *EMNLP-IJCNLP*.
- John H. Jenkins, Richard Cook, and Ken Lunde. 2019. Uax #38: Unicode han database (uni-han). <http://www.unicode.org/reports/tr38/tr38-27.html>.
- Karthikeyan K, Zihan Wang, Stephen Mayhew, and Dan Roth. 2020. Cross-lingual ability of multilingual BERT: An empirical study. In *ICLR*.
- Nobuhiro Kaji and Masaru Kitsuregawa. 2014. Accurate word segmentation and POS tagging for japanese microblogs: Corpus annotation and joint modeling with lexical normalization. In *EMNLP*.
- Daisuke Kawahara, Yuichiro Machida, Tomohide Shiba, Sadao Kurohashi, Hayato Kobayashi, and Manabu Sassano. 2014. Rapid development of a corpus with discourse annotations using two-stage crowdsourcing. In *COLING*.

- Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.
- Yoshiaki Kitagawa and Mamoru Komachi. 2018. Long short-term memory for japanese word segmentation. In *PACLIC*.
- Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M. Rush. 2017. OpenNMT: Open-source toolkit for neural machine translation. In *ACL*.
- Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. In *NeurIPS*.
- Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc’Aurelio Ranzato. 2018a. Unsupervised machine translation using monolingual corpora only. In *ICLR*.
- Guillaume Lample, Alexis Conneau, Marc’Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018b. Word translation without parallel data. In *ICLR*.
- Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc’Aurelio Ranzato. 2018c. Phrase-based & neural unsupervised machine translation. In *EMNLP*.
- Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *ICLR*. OpenReview.net.
- Patrick S. H. Lewis, Barlas Oguz, Ruty Rinott, Sebastian Riedel, and Holger Schwenk. 2019. MLQA: evaluating cross-lingual extractive question answering. *CoRR*, abs/1910.07475.
- Xiaoya Li, Yuxian Meng, Xiaofei Sun, Qinghong Han, Arianna Yuan, and Jiwei Li. 2019. Is word segmentation necessary for deep learning of chinese representations? In *ACL*.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.
- Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Bilingual word representations with monolingual quality in mind. In *VS@HLT-NAACL*.
- Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013. Exploiting similarities among languages for machine translation. *CoRR*, abs/1309.4168.
- Laura Miller. 1998. Wasei eigo: English “loanwords” coined in japan. *The life of language: Papers in linguistics in honor of William Bright*, pages 123–139.
- RobertoNavigli, José Camacho-Collados, and Alessandro Raganato. 2017. Word sense disambiguation: A unified evaluation framework and empirical comparison. In *EACL*.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*.
- Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL-HLT*.
- Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual bert? In *ACL*. Association for Computational Linguistics.
- Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *ACL*.
- Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.
- Chunqi Wang and Bo Xu. 2017. Convolutional neural network with word embeddings for chinese word segmentation. In *IJCNLP*.
- Shijie Wu and Mark Dredze. 2019. Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT. In *EMNLP-IJCNLP*.
- Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google’s neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.
- Canwen Xu, Feiyang Wang, Jialong Han, and Chenliang Li. 2019. Exploiting multiple embeddings for chinese named entity recognition. In *CIKM*.
- Jie Yang, Yue Zhang, and Fei Dong. 2017. Neural word segmentation with rich pretraining. In *ACL*.
- Jie Yang, Yue Zhang, and Shuailong Liang. 2019a. Subword encoding in lattice LSTM for chinese word segmentation. In *NAACL-HLT*.
- Yinfei Yang, Yuan Zhang, Chris Tar, and Jason Baldridge. 2019b. PAWS-X: A cross-lingual adversarial dataset for paraphrase identification. In *EMNLP-IJCNLP*.
- Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019c. Xlnet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*.

Longtu Zhang and Mamoru Komachi. 2019. Chinese-japanese unsupervised neural machine translation using sub-character level information. In *The 33rd Pacific Asia Conference on Language, Information and Computation*.

Meng Zhang, Yang Liu, Huanbo Luan, and Maosong Sun. 2017. Earth mover’s distance minimization for unsupervised bilingual lexicon induction. In *EMNLP*.

Hao Zhou, Zhenting Yu, Yue Zhang, Shujian Huang, Xin-Yu Dai, and Jiajun Chen. 2017. Word-context character embeddings for chinese word segmentation. In *EMNLP*.

Pierre Zweigenbaum, Serge Sharoff, and Reinhard Rapp. 2017. Overview of the second BUCC shared task: Spotting parallel sentences in comparable corpora. In *BUCC@ACL*.

# Towards a Better Understanding of Label Smoothing in Neural Machine Translation

Yingbo Gao Weiyue Wang Christian Herold Zijian Yang Hermann Ney

Human Language Technology and Pattern Recognition Group

Computer Science Department

RWTH Aachen University

D-52056 Aachen, Germany

{gao|wwang|herold|zyang|ney}@i6.informatik.rwth-aachen.de

## Abstract

In order to combat overfitting and in pursuit of better generalization, label smoothing is widely applied in modern neural machine translation systems. The core idea is to penalize over-confident outputs and regularize the model so that its outputs do not diverge too much from some prior distribution. While training perplexity generally gets worse, label smoothing is found to consistently improve test performance. In this work, we aim to better understand label smoothing in the context of neural machine translation. Theoretically, we derive and explain exactly what label smoothing is optimizing for. Practically, we conduct extensive experiments by varying which tokens to smooth, tuning the probability mass to be deducted from the true targets and considering different prior distributions. We show that label smoothing is theoretically well-motivated, and by carefully choosing hyperparameters, the practical performance of strong neural machine translation systems can be further improved.

## 1 Introduction

In recent years, Neural Network (NN) models bring steady and concrete improvements on the task of Machine Translation (MT). From the introduction of sequence-to-sequence models (Cho et al., 2014; Sutskever et al., 2014a), to the invention of the attention mechanism (Bahdanau et al., 2015; Luong et al., 2015), end-to-end sequence learning with attention becomes the dominant design choice for Neural Machine Translation (NMT) models. From the study of convolutional sequence to sequence learning (Gehring et al., 2017a,b), to the prosperity of self-attention networks (Vaswani et al., 2017; Devlin et al., 2019), modern NMT systems, especially Transformer-based ones (Vaswani et al., 2017), often deliver state-of-the-art performances

(Bojar et al., 2018; Barrault et al., 2019), even under the condition of large-scale corpora (Ott et al., 2018; Edunov et al., 2018).

In Transformer-based models, label smoothing is a widely applied method to improve model performance. Szegedy et al. (2016) initially introduce the method when making refinements to the Inception (Szegedy et al., 2015) model, with the motivation to combat overfitting and improve adaptability. In principle, label smoothing discounts a certain probability mass from the true label and redistributes it uniformly across all the class labels. This lowers the difference between the largest probability output and the others, effectively discouraging the model to generate overly confident predictions. Since information entropy (Shannon, 1948) can be thought of as a confidence measure of a probability distribution, Pereyra et al. (2017) add a negative entropy regularization term to the conventional cross entropy training criterion and compare it with uniform smoothing and unigram smoothing. Müller et al. (2019) deliver further insightful discussions about label smoothing, empirically investigating it in terms of model calibration, knowledge distillation and representation learning.

Label smoothing itself is an interesting topic that brings insights about the general learnability of a neural model. While existing methods are rather heuristical in their nature, the fact that simply discounting some probability mass from the true label and redistributing it with some prior distribution (see Figure 1 for an illustration) works in practice, is worth to be better understood.

In this paper, we raise two high-level research questions to outline our work:

1. Theoretically, what is label smoothing (or the related confidence penalty) optimizing for?
2. Practically, what is a good recipe in order to apply label smoothing successfully in NMT?

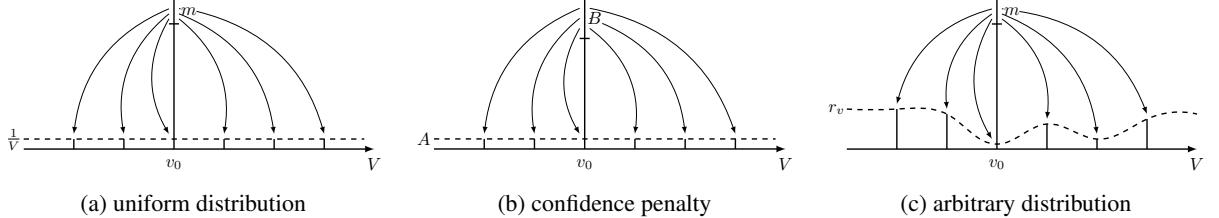


Figure 1: An illustration of label smoothing with various prior distributions.  $m$  and  $B$  are discounted probability masses.  $V$  is the vocabulary size and  $v_0$  is the correct target word.  $\frac{1}{V}$ ,  $A$  and  $r_v$  are prior distributions. Smoothing with (a),  $m$  is equally redistributed across the vocabulary. Smoothing with (b),  $A$  is implicitly  $\frac{1}{V}$  everywhere as well, and the exact value of  $B$  can be obtained (Section 3.2). Smoothing with (c),  $m$  goes to each class in proportion to an arbitrary smoothing prior  $r_v$  (Section 4.3).

The presentation of our results is organized into three major sections:

- First, we introduce a generalized formula for label smoothing and derive the theoretical solution to the training problem.
- Second, we investigate various aspects that affect the training process and show an empirically good recipe to apply label smoothing.
- Finally, we examine the implications in search and scoring and motivate further research into the mismatch between training and testing.

## 2 Related Work

The extensive use of NNs in MT (Bojar et al., 2016, 2017, 2018; Barrault et al., 2019) is a result of many pioneering and inspiring works. Continuous-valued word vectors lay the foundation of modern Natural Language Processing (NLP) NNs, capturing semantic and syntactic relations and providing numerical ways to calculate meaningful distances among words (Bengio et al., 2001; Schwenk et al., 2006; Schwenk, 2007; Sundermeyer et al., 2012; Mikolov et al., 2013a,b). The investigations of sequence-to-sequence learning (Cho et al., 2014; Sutskever et al., 2014b), the studies of attention mechanism (Bahdanau et al., 2015; Luong et al., 2015) and the explorations into convolutional and self-attention NNs (Gehring et al., 2017a,b; Vaswani et al., 2017) mark steady and important steps in the field of NMT. Since the introduction of BERT (Devlin et al., 2019), the Transformer model (Vaswani et al., 2017) becomes the de facto architectural choice for many competitive NLP systems. Among the numerous ingredients that make Transformer networks successful, label smoothing is one that must not be overlooked and shall be the focus of this work.

The idea of smoothing is not new in itself. For instance, many smoothing heuristics and functions are investigated in the context of count-based language modeling (Jelinek and Mercer, 1980; Katz, 1987; Church and Gale, 1991; Kneser and Ney, 1995; Chen and Goodman, 1996). Interestingly, when training NNs, the idea of smoothing comes in a new form and is applied on the empirical one-hot target distributions.

Proposed to counteract overfitting and pursue better generalization, label smoothing (Szegedy et al., 2016) finds its first applications in NNs in the field of computer vision. Later, the method is shown to be effective in MT (Vaswani et al., 2017). Furthermore, it is also helpful when applied in other scenarios, e.g. Generative Adversarial Networks (GANs) (Salimans et al., 2016), automatic speech recognition (Chiu et al., 2018), and person re-identification (Ainam et al., 2019). Since the method centralizes on the idea of avoiding over-confident model outputs on training data, it is reanalyzed in Pereyra et al. (2017). The authors include an additional confidence penalty regularization term in the training loss, and compare it to standard label smoothing with uniform or unigram prior. While label smoothing boosts performance significantly compared to using hard target labels, the difference in performance gains when comparing different smoothing methods is relatively small. Müller et al. (2019) bring recent advancements towards better intuitive understandings of label smoothing. They observe a clustering effect of learned features and argue that label smoothing improves model calibration, yet hurting knowledge distillation when the model is used as a teacher for another student network.

As a regularization technique in training, label smoothing can be compared against other methods such as dropout (Srivastava et al., 2014) and Dis-

turbLabel (Xie et al., 2016). Intuitively, dropout can be viewed as ensembling different model architectures on the same data and DisturbLabel can be viewed as ensembling the same model architecture on different data, as pointed out in Xie et al. (2016). Interestingly, label smoothing can also be understood as estimating the marginalized label dropout during training (Pereyra et al., 2017). In this paper, we propose two straightforward extensions to label smoothing, examining token selection and prior distribution. Salimans et al. (2016) and Zhou et al. (2017) investigate a similar issue to the former. In the context of GANs, they select only those positive examples to smooth while we consider the task of MT, discussing how many tokens to smooth and how they should be selected. Pereyra et al. (2017) and Gao et al. (2019) talk about ideas similar to the latter. In their respective contexts, one experiments with unigram probabilities for label smoothing and the other uses Language Model (LM) posteriors to softly augment the source and target side of MT training data.

### 3 Solving the Training Problem

The standard label smoothing (STN) loss, as used by Vaswani et al. (2017), can be expressed as:

$$L^{\text{STN}} = - \sum_{n=1}^N \sum_{v=1}^V \left( (1-m)p_v + m\frac{1}{V} \right) \log q_v \quad (1)$$

where  $L^{\text{STN}}$  denotes the cross entropy with standard label smoothing,  $n$  is a running index in the total number of training tokens  $N$ ,  $v$  is a running index in the target vocabulary  $V$ ,  $m$  is the hyperparameter that controls the amount of probability mass to discount,  $p_v$  is the one-hot true target distribution and  $q_v$  is the output distribution of the model.

The confidence penalty (CFD) loss, as used by Pereyra et al. (2017), can be expressed as:

$$L^{\text{CFD}} = - \sum_{n=1}^N \sum_{v=1}^V (p_v - m'q_v) \log q_v \quad (2)$$

where  $L^{\text{CFD}}$  denotes the confidence-penalized cross entropy,  $m'$  in this case is the hyperparameter that controls the strength of the confidence penalty and thus differs from  $m$  in Equation 1.

In both cases, the outer summation is over all of the training tokens  $N$ , implicating that all of the target token probabilities are smoothed. The

dependencies of  $q_v$  and  $p_v$  on  $n$  are omitted for simplicity.

Additionally for Equation 1, authors of both papers (Vaswani et al., 2017; Pereyra et al., 2017) point out that the uniform prior can be replaced with alternative distributions over the target vocabulary. One more thing to notice is the negative sign in front of the non-negative term  $m'$  in Equation 2, which means that  $p_v - m'q_v$  is not a probability distribution anymore. One can nonetheless apply tricks to normalize the term inside the parentheses so that it becomes a probability distribution, e.g.:

$$L_{\text{normalized}_1}^{\text{CFD}} = - \sum_{n=1}^N \sum_{v=1}^V \log q_v \cdot \frac{(p_v - m'q_v) - \min(p_v - m'q_v)}{\sum_{v'=1}^V (p_{v'} - m'q_{v'}) - \min(p_{v'} - m'q_{v'})} \quad (3)$$

or

$$L_{\text{normalized}_2}^{\text{CFD}} = - \sum_{n=1}^N \sum_{v=1}^V \log q_v \cdot \frac{\exp(p_v - m'q_v)}{\sum_{v'=1}^V \exp(p_{v'} - m'q_{v'})} \quad (4)$$

and implement it as an additional layer of activation during training, where  $v'$  is an alternative running index in the vocabulary. In any case, the integration of Equation 2 into the form of Equation 1 cannot be done without significantly modifying the original confidence penalty, and we leave it for future work.

#### 3.1 Generalized Formula

In an effort to obtain a unified view, we propose a simple generalized formula and make two major changes. First, we separate the outer summation over the tokens and divide it into two summations, namely “not to smooth” and “to smooth”. Second, we modify the prior distribution to allow it to depend on the position, current token and model output. In this case,  $r$  could be the posterior from some helper model (e.g. an LM), and during training, obtaining it on-the-fly is not expensive, as previously shown (Bi et al., 2019; Wang et al., 2019). The generalized label smoothing (GNR) loss can be expressed as:

$$L^{\text{GNR}} = - \sum_{n \in \mathcal{A}} \sum_{v=1}^V p_v \log q_v - \sum_{n \in \mathcal{B}} \sum_{v=1}^V ((1-m)p_v + mr_{v,q_v}) \log q_v \quad (5)$$

where  $L^{\text{GNR}}$  denotes the generalized cross entropy,  $\mathcal{A}$  is the set of tokens not to smooth,  $\mathcal{B}$  is the set of tokens to smooth,  $r_{v,q_v}$  is an arbitrary prior distribution for smoothing and again we drop the dependencies of  $p_v$ ,  $q_v$  and  $r_{v,q_v}$  on  $n$  for simplicity.

A natural question when explicitly writing out  $\mathcal{A}$  and  $\mathcal{B}$ , s.t.  $\mathcal{A} \cap \mathcal{B} = \emptyset$  and  $|\mathcal{A} \cup \mathcal{B}| = N$ , is which tokens to include in  $\mathcal{B}$ . Here, we consider two simple ideas: uniform random sampling (RND) and an entropy-based uncertainty heuristic (ENT). The former chooses a certain percentage of tokens to smooth by sampling tokens uniformly at random. The latter prioritizes those tokens whose prior distributions have higher entropy. The logic behind the ENT formulation is that when the prior distribution is flattened out, yielding a higher entropy, the helper model is uncertain about the current position, and the model output should thus be smoothed. Formally, the two heuristics can be expressed as:

$$\mathcal{B}^{\text{RND}} = \{n; \rho_n \sim U(0, 1), \rho_n \leq \pi\} \quad (6)$$

$$\mathcal{B}^{\text{ENT}} = \{b_1, b_2, \dots, b_{\lceil \pi N \rceil}\} \quad (7)$$

where  $\rho_n$  is a sample from the uniform distribution  $U$  in  $[0, 1]$ ,  $\pi$  is a hyperparameter controlling the percentage of tokens to smooth and  $\{b_1, b_2, \dots, b_N\}$  is a permutation of data indices  $\{1, 2, \dots, N\}$  in descending order of the entropy of prior  $r$ , i.e.  $\forall 1 \leq i \leq j \leq N, -\sum_V r_{b_i} \log r_{b_i} \geq -\sum_V r_{b_j} \log r_{b_j}$ .

The hyperparameter  $m$  in Equation 5 deserves some further notice. This is essentially the parameter that controls the strength of the label smoothing procedure. When it is zero, no smoothing is done. When it is one and  $|\mathcal{B}| = N$ , the model is optimized to output the prior distribution  $r$ . One can obviously further generalize it so that  $m$  depends also on  $n, v$  and  $q_v$ . However in this work, we focus on the outer summation in  $N$  and alternative priors  $r$ , and leave the exploration of adaptive smoothing strength  $m_{n,r,q_v}$  for future work.

### 3.2 Theoretical Solution

When it comes to the analysis of label smoothing, previous works focus primarily on intuitive understandings. Pereyra et al. (2017) observe that both label smoothing and confidence penalty lead to smaller gradient norms during training. Müller et al. (2019) argue that label smoothing helps beam-search by improving model calibration. They further visualize the learned features and show a clustering effect of features from the same class. In this work, we concentrate on finding a theoretical

solution to the training problem, and show exactly what label smoothing and confidence penalty are optimizing for.

Consider the optimization problem when training with Equation 1:

$$\min_{q_1, q_2, \dots, q_V} L_n^{\text{STN}}, \quad \text{s.t. } \sum_{v=1}^V q_v = 1 \quad (8)$$

While in practice we use gradient optimizers to obtain a good set of parameters of the NN, the optimization problem actually has well-defined analytical solutions locally:

$$\tilde{q}_v^{\text{STN}} = (1 - m)p_v + m \frac{1}{V} \quad (9)$$

which is simply a linear interpolation between the one-hot target distribution  $p_v$  and the smoothing prior  $\frac{1}{V}$ , with  $m \in [0, 1]$  being the interpolation weight. One can use either the divergence inequality or the Lagrange multiplier method to obtain this result (see Appendix A).

Consider the optimization problem when training with Equation 2:

$$\min_{q_1, q_2, \dots, q_V} L_n^{\text{CFD}}, \quad \text{s.t. } \sum_{v=1}^V q_v = 1 \quad (10)$$

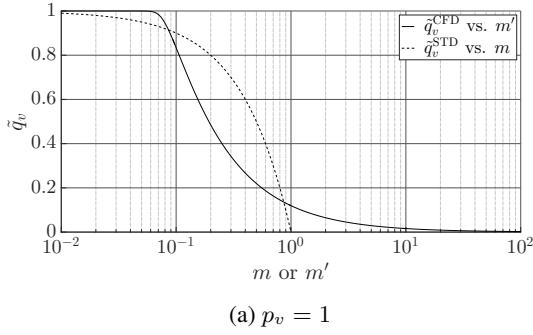
The problem becomes harder because now the regularization term also depends on  $q_v$ . Introducing the Lagrange multiplier  $\lambda$  and solving for optima will result in a transcendental equation. Making use of the Lambert  $W$  function (Corless et al., 1996), the solution can be expressed as (see Appendix A for detailed derivation):

$$\tilde{q}_v^{\text{CFD}} = \frac{p_v}{m' W_0 \left( \frac{p_v}{m'} e^{1 + \frac{\lambda}{m'}} \right)} \quad (11)$$

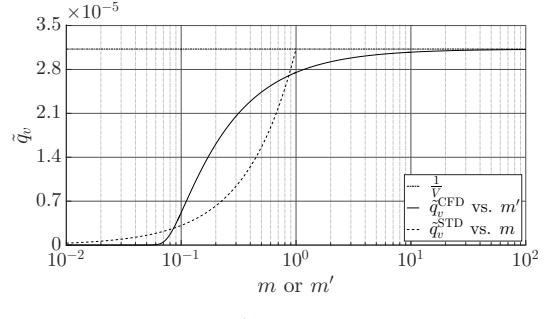
where  $W_0$  is the principal branch of the Lambert  $W$  function and  $\lambda$  is the Lagrange multiplier, which is numerically solvable<sup>1</sup> when non-negative  $m'$  and probability distribution  $p_v$  are given. Equation 11 essentially gives a non-linear relationship between  $\tilde{q}_v^{\text{CFD}}$  and  $p_v$ , controlled by the hyperparameter  $m'$ .

Now that theoretical solutions are presented in Equation 9 and 11, it is possible to plot the graphs of optimal  $\tilde{q}_v$ , with respect to  $m$  and  $m'$ . Shown in Figure 2, as expected for both STN and CFD, the overall effect is to decrease  $q_v$  when  $p_v = 1$  and increase  $q_v$  when  $p_v = 0$ . When  $m$  or  $m'$  gets large

<sup>1</sup>One can use  $\lim_{m' \rightarrow 0} \tilde{q}_v^{\text{CFD}}$  to avoid division by zero.



(a)  $p_v = 1$



(b)  $p_v = 0$

Figure 2: Graphs of optimal  $\tilde{q}_v$  w.r.t.  $m$  or  $m'$ . Note the logarithmic scale in horizontal axes, with  $m \in [0, 1]$  and  $m' \geq 0$ . In order to obtain numerical solutions for  $\tilde{q}_v^{\text{STD}}$  and  $\tilde{q}_v^{\text{CFD}}$ , we set  $V = 32000$ , which is a common vocabulary size when operating on sub-word levels.

enough, the total probability mass is discounted and  $\frac{1}{V}$  is redistributed to each token in the vocabulary. The graph of GRN<sup>2</sup> is similar to STD, only changing the limit from  $\frac{1}{V}$  to  $r_v$  as  $m$  approaches one, and not included here for brevity. One last thing to notice is that the outer summation over the tokens is ignored. If it is taken into consideration,  $\tilde{q}$  is dragged towards the empirical distribution given by the corpus<sup>3</sup>.

## 4 Finding a Good Recipe

In this section, we describe our results and insights towards a good recipe to successfully apply label smoothing. We experiment with six IWSLT2014 datasets: German (de), Spanish (es), Italian (it), Dutch (nl), Romanian (ro), Russian (ru) to English (en), and one WMT2014 dataset: English to German. The statistics of these datasets are summarized in Table 1. To prepare the subword tokens, we adopt joint byte pair encoding (Sennrich et al., 2016), and use 10K and 32K merge operations on IWSLT and WMT, respectively. When preprocessing IWSLT, we remove sentences longer than 175 words, lowercase both source and target sides, randomly subsample roughly 4.35% of the training sentence pairs as development data and concatenate all previously available development and test sets as test data, similar to Gehring et al. (2017a). As for the preprocessing of WMT, we follow the setup in Ott et al. (2018). Using the Transformer architec-

ture (Vaswani et al., 2017), we apply the base setup for IWSLT and the big setup for WMT. For all language pairs, we share all three embedding matrices. All helper models are also Transformer-based. We conduct all experiments using fairseq (Ott et al., 2019), monitor development set perplexity during training, and report BLEU (Papineni et al., 2002) scores on test sets after beam search.

### 4.1 Token Selection

The first thing to determine is how to select tokens for smoothing and how many tokens to smooth.

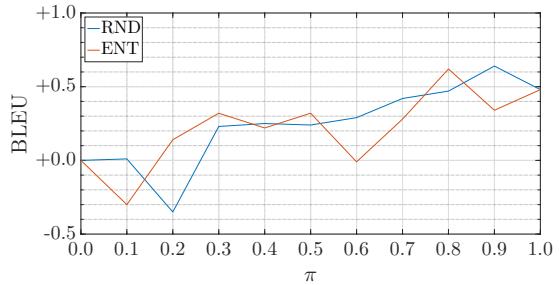


Figure 3: Smoothing with RND versus ENT on de-en.  $m$  is set to 0.1. The development and test perplexities of the helper LM are 53.8 and 46.5.

For this purpose, we begin by considering models smoothed with an LM helper. The helper LM is trained on target sentences from the corresponding parallel data till convergence. Figure 3 shows a comparison between RND and ENT, varying the percentage of smoothed tokens  $\pi$  and using the absolute performance improvements in BLEU as the vertical axis. Since the two methods only affect the order in which tokens are selected, they should yield the exact same results when all tokens are selected. This can be clearly seen from the figure and serves as a sanity check for the correctness of

<sup>2</sup>Assuming  $r$  only depends on  $n, v$  and not  $q_v$ . In the latter case, one needs to solve the optimization problem ignoring the outer summation and reusing the Lagrange multiplier.

<sup>3</sup>For an intuitive understanding, consider the case when two sentence pairs have the exact same context up to a certain target position but the next tokens are different (e.g. “Danke .” in German being translated to “Thank you .” and “Thank you very much .” in English, the period in the first translation and “very” in the second translation have the same context.)

dataset		IWSLT						WMT
language pair		de-en	es-en	it-en	nl-en	ro-en	ru-en	en-de
number of sentence pairs	train	160K	169K	167K	154K	168K	153K	4.50M
	valid	7.3K	7.7K	7.6K	7.0K	7.6K	7.0K	3.0K
	test	6.8K	5.6K	6.6K	5.4K	5.6K	5.5K	3.0K

Table 1: Data statistics of IWSLT and WMT datasets.

the implementation. The RND and ENT curves follow a similar trend, increasing with the number of smoothed tokens. From the curves, neither selection method is consistently better than the other, indicating that the entropy-based selection heuristics is probably an oversimplification considering the stochasticity introduced when altering the number of smoothed tokens. We continue to examine the uphill trend seen in Figure 3 in other cases.

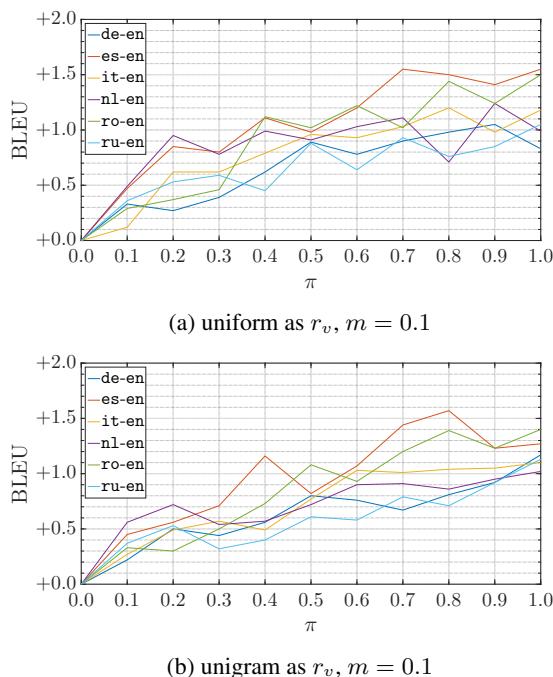


Figure 4: Smoothing different percentages of tokens.

Figure 4 reveals the relationship between absolute BLEU improvements and  $\pi$ , when smoothing with uniform or unigram (RND) distributions. While for each language pair the actual changes in BLEU differ, it is clear to conclude that, the more tokens smoothed, the better the performance. This conclusion is rather universal and holds true for the majority of our experiment settings (varying  $m$  and  $r$ ). From here on, we smooth all tokens, i.e.  $|\mathcal{B}| = N$ , by default.

## 4.2 Probability Mass

Our next goal is to find good values of  $m$ .

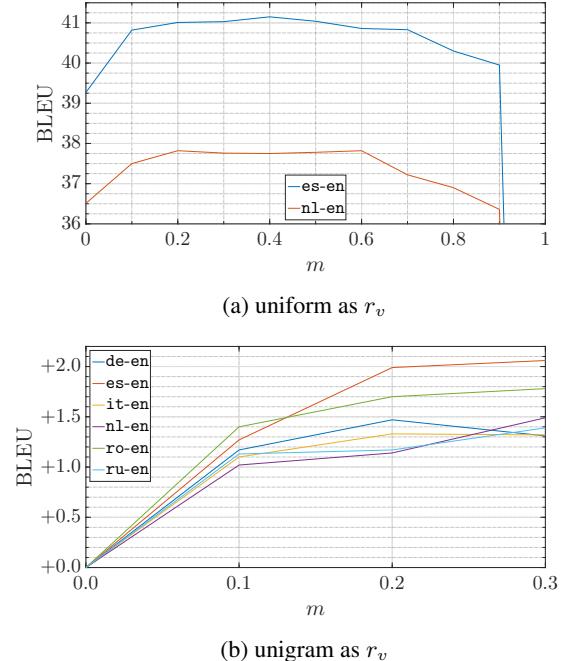


Figure 5: Discounting different probability masses.

The discounted probability mass  $m$  is a tunable hyperparameter that is set to 0.1 in the original Transformer (Vaswani et al., 2017) paper. We vary this parameter in the case of uniform smoothing and unigram smoothing, and plot the results in Figure 5. As shown in Figure 5a, the BLEU score immediately improves at  $m = 0.1$ , then plateaus when  $m \in [0.3, 0.6]$ , slowly decreases when  $m \in [0.7, 0.9]$  and quickly drops to zero when  $m$  approaches one. When  $m = 1$ , the model is optimized towards a uniform distribution and completely ignores the training data. Because perplexity can be thought of as the effective vocabulary size of a model, we examine the perplexities when  $m = 1$  for both language pairs. As expected, the development perplexities are around 10K, which is in the same order of magnitude as the corresponding vocabulary sizes. Another interesting observation is that the BLEU scores only drop when

$m$  gets close to one and the model produces acceptable translations elsewhere. This indicates that NN models trained with gradient optimizers are very good at picking out the effective training signals even when they are buried in much stronger noise signals (the uniform smoothing priors in the case of Figure 5a). This could be further related to multi-task learning (Ruder, 2017), where the system performances are also related to the regularization weights of the auxiliary losses. For unigram, we vary  $m$  in  $\{0.1, 0.2, 0.3\}$ . As seen in Figure 5b, while smoothing with  $m = 0.1$  gives a large improvement over no smoothing, setting  $m = 0.3$  further boosts the performance, consistently for all six IWSLT language pairs.

### 4.3 Prior Distribution

Furthermore, we explore the use of LM and MT posteriors as prior distributions for smoothing.

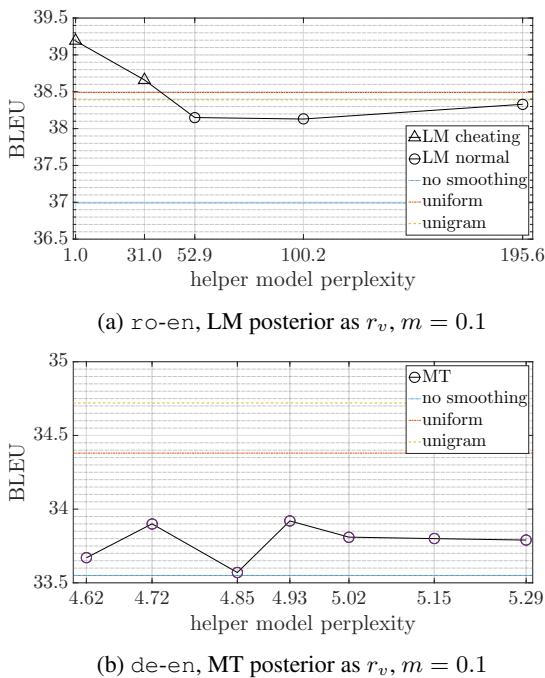


Figure 6: Smoothing with LM and MT posteriors.

We train systems using Transformer LMs and MT models of different qualities for label smoothing, as in Figure 6. To obtain very good LMs, we train them with test data and mark the cheating LMs in Figure 6a. We additionally plot the BLEU scores of models with no smoothing, smoothed with uniform and unigram, as horizontal lines to compare the absolute performances. Intuitively, the curve should follow a downhill trend, meaning that the worse the helper model performs, the worse the

model smoothed with it performs. This is loosely the case for LM, with cheating LMs giving better performances than uniform and unigram, and normal LMs lacking behind. As for MT, improvement over the no smoothing case is seen in Figure 6b. However, neither the downhill trend nor the competence over other priors in terms of BLEU, is seen. This suggests that the model is probably not utilizing the information in the soft distribution effectively. Related to knowledge distillation (Hinton et al., 2015; Kim and Rush, 2016), a trainable teacher (the helper model in our case) might be further beneficial (Bi et al., 2019; Wang et al., 2018).

One important thing to mention is that, while neither LM nor MT outperforms uniform or unigram in terms of test BLEU score in our experiments, we see significant drops in development set perplexities when smoothing with LM or MT. This signals a mismatch between training and testing, and suggests that smoothing with LM or MT indeed works well for the optimization criterion, but not as much for the final metric, the calculation of which involves beam search and scoring of the discrete tokens.

### 4.4 Final Results

Finally, we report BLEU scores of our best systems across all language pairs in Table 2. While applying uniform label smoothing significantly improves over the baselines, by using a good recipe, an additional improvement of around +0.5 BLEU is obtained across all language pairs. For the hyperparameters, we find that smoothing **all tokens** by  **$m = 0.3$  with a unigram prior** is a good recipe, consistently giving one of the best BLEU scores.

## 5 Analyzing the Mismatch

As discussed in Section 4.3, models smoothed with LMs or MT model posteriors yield very good development set perplexities but no big improvements in terms of test BLEU scores. Here, we further investigate this phenomenon in terms of search and scoring.

### 5.1 Search

We first plot the test BLEU scores with respect to the beam size used during search. In Figure 7, we see that the dashed curves for “no smoothing”, “uniform” and “unigram” initially increase and then plateau, which is an expected shape (see Figure 8

dataset	IWSLT						WMT
language pair	de-en	es-en	it-en	nl-en	ro-en	ru-en	en-de
no label smoothing	33.6	39.3	31.2	36.5	37.0	22.3	28.0
Vaswani et al. (2017)	34.4	40.8	32.4	37.5	38.5	23.4	28.4
our best recipe	<b>35.0</b>	<b>41.5</b>	<b>32.8</b>	<b>38.0</b>	<b>39.0</b>	<b>23.9</b>	<b>29.0</b>

Table 2: BLEU scores can be significantly improved with good label smoothing recipes. The first row of numbers corresponds to using only the cross entropy criterion for training. The second row of numbers corresponds to the Transformer baselines. The last row contains scores obtained with our best hyperparameters.

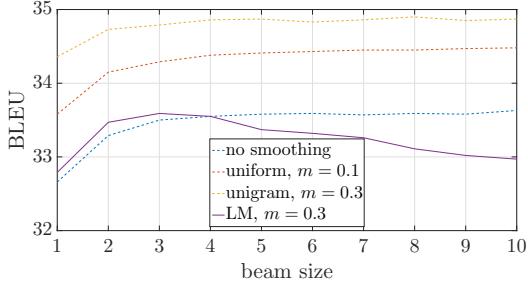
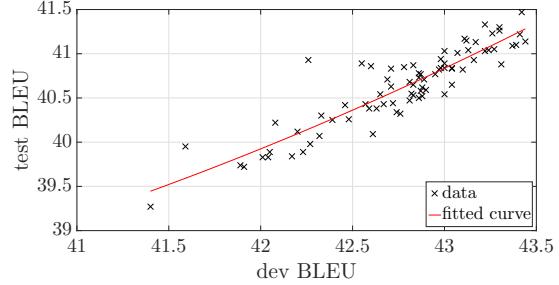


Figure 7: BLEU versus beam size on de-en.

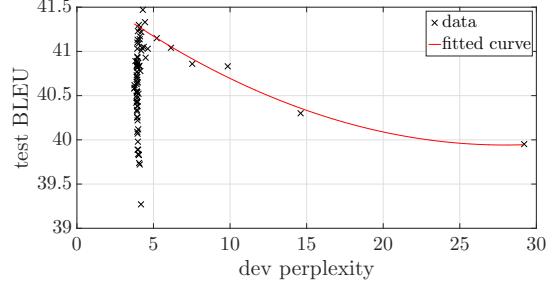
in Zhou et al. (2019)). However, the solid curve for LM drops quickly as beam size increases (see Stahlberg and Byrne (2019) for more insight). A possible explanation is that models smoothed with LMs generate search spaces that are richer in probability variations and more diversified, compared to e.g. uniform label smoothing. As search becomes stronger, hypotheses that have higher probabilities, but not necessarily closer to the true targets, are found. This suggests that the mismatch in development set perplexity and test BLEU is a complex phenomenon and calls for more analysis.

## 5.2 Scoring

We further examine test BLEU with respect to development (dev) BLEU and dev perplexity. As shown in Figure 8a, test BLEU is nicely correlated with dev BLEU, indicating that there is no mismatch between dev and test in the dataset itself. However, as in Figure 8b, although test BLEU increases with a decreasing dev perplexity, in regions of low dev perplexities, there exist many systems with very different test performances ranging from 39.3 BLEU to 41.5 BLEU. Despite perplexity being directly related to the cross entropy training criterion, this is an example where it fails to be a good proxy for the final BLEU metric. Against this mismatch between training and testing, either a more BLEU-related dev score or a more perplexity-related test metric needs to be considered.



(a) Dev BLEU is a good proxy for test BLEU.



(b) Dev perplexity is a bad proxy for test BLEU.

Figure 8: Relationships between test BLEU and dev metrics. 79 converged es-en models with different label smoothing hyperparameters are scattered.

## 6 Conclusion

In this work, we investigate label smoothing in neural machine translation. Considering important aspects in label smoothing: token selection, probability mass and prior distribution, we introduce a generalized formula and derive theoretical solutions to the training problem. Examining the effect of various hyperparameter choices, practically we show that with a good label smoothing recipe, one can obtain consistent improvements over strong baselines. Delving into search and scoring, we finally emphasize the mismatch between training and testing, and motivate future research. Reassuring that label smoothing brings concrete improvements and considering that it only operates at the output side of the model, our next step is to explore similar smoothing ideas at the input side.

## Acknowledgements



This work has received funding from the European Research Council (ERC) (under the European Union’s Horizon 2020 research and innovation programme, grant agreement No 694537, project “SEQCLAS”) and the Deutsche Forschungsgemeinschaft (DFG; grant agreement NE 572/8-1, project “CoreTec”). The GPU computing cluster was supported by DFG (Deutsche Forschungsgemeinschaft) under grant INST 222/1168-1 FUGG.

## References

- J. Ainam, K. Qin, G. Liu, and G. Luo. 2019. [Sparse label smoothing regularization for person re-identification](#). *IEEE Access*, 7:27899–27910.
- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. [Neural machine translation by jointly learning to align and translate](#). In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.
- Loïc Barrault, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Müller, Santanu Pal, Matt Post, and Marcos Zampieri. 2019. [Findings of the 2019 conference on machine translation \(WMT19\)](#). In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 1–61, Florence, Italy. Association for Computational Linguistics.
- Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. 2001. [A neural probabilistic language model](#). In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 932–938. MIT Press.
- tianchi Bi, hao xiong, Zhongjun He, Hua Wu, and Haifeng Wang. 2019. [Multi-agent learning for neural machine translation](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 856–865, Hong Kong, China. Association for Computational Linguistics.
- Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Shujian Huang, Matthias Huck, Philipp Koehn, Qun Liu, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Raphael Rubino, Lucia Specia, and Marco Turchi. 2017. [Findings of the 2017 conference on machine translation \(WMT17\)](#). In *Proceedings of the Second Conference on Machine Translation*, pages 169–214, Copenhagen, Denmark. Association for Computational Linguistics.
- Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Aurélie Névéol, Mariana Neves, Martin Popel, Matt Post, Raphael Rubino, Carolina Scarton, Lucia Specia, Marco Turchi, Karin Verspoor, and Marcos Zampieri. 2016. [Findings of the 2016 conference on machine translation](#). In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 131–198, Berlin, Germany. Association for Computational Linguistics.
- Ondřej Bojar, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Philipp Koehn, and Christof Monz. 2018. [Findings of the 2018 conference on machine translation \(WMT18\)](#). In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 272–303, Belgium, Brussels. Association for Computational Linguistics.
- Stanley F. Chen and Joshua Goodman. 1996. [An empirical study of smoothing techniques for language modeling](#). In *34th Annual Meeting of the Association for Computational Linguistics*, pages 310–318, Santa Cruz, California, USA. Association for Computational Linguistics.
- C. Chiu, T. N. Sainath, Y. Wu, R. Prabhavalkar, P. Nguyen, Z. Chen, A. Kannan, R. J. Weiss, K. Rao, E. Gonina, N. Jaitly, B. Li, J. Chorowski, and M. Bacchiani. 2018. [State-of-the-art speech recognition with sequence-to-sequence models](#). In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4774–4778.
- Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. [Learning phrase representations using RNN encoder–decoder for statistical machine translation](#). In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.
- Kenneth W. Church and William A. Gale. 1991. A comparison of the enhanced Good-Turing and deleted estimation methods for estimating probabilities of English bigrams. *Computer Speech and Language*, 5:19–54.
- Robert M Corless, Gaston H Gonnet, David EG Hare, David J Jeffrey, and Donald E Knuth. 1996. On the lambertw function. *Advances in Computational mathematics*, 5(1):329–359.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. [BERT: Pre-training of](#)

deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. [Understanding back-translation at scale](#). In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium. Association for Computational Linguistics.

Fei Gao, Jinhua Zhu, Lijun Wu, Yingce Xia, Tao Qin, Xueqi Cheng, Wengang Zhou, and Tie-Yan Liu. 2019. [Soft contextual data augmentation for neural machine translation](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5539–5544, Florence, Italy. Association for Computational Linguistics.

Jonas Gehring, Michael Auli, David Grangier, and Yann Dauphin. 2017a. [A convolutional encoder model for neural machine translation](#). In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 123–135, Vancouver, Canada. Association for Computational Linguistics.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017b. [Convolutional sequence to sequence learning](#). In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1243–1252, International Convention Centre, Sydney, Australia. PMLR.

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. [Distilling the knowledge in a neural network](#). *CoRR*, abs/1503.02531.

Fred Jelinek and Robert L. Mercer. 1980. Interpolated estimation of Markov source parameters from sparse data. In Edzard S. Gelsema and Laveen N. Kanal, editors, *Proceedings, Workshop on Pattern Recognition in Practice*, pages 381–397. North Holland, Amsterdam.

Slava M. Katz. 1987. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Trans. Acoustics, Speech, and Signal Processing*, 35:400–401.

Yoon Kim and Alexander M. Rush. 2016. [Sequence-level knowledge distillation](#). In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327, Austin, Texas. Association for Computational Linguistics.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 181–184, Detroit, Michigan, USA.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. [Effective approaches to attention-based neural machine translation](#). In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. [Distributed representations of words and phrases and their compositionality](#). In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2, NIPS’13*, pages 3111–3119, USA. Curran Associates Inc.

Rafael Müller, Simon Kornblith, and Geoffrey E. Hinton. 2019. [When does label smoothing help?](#) *CoRR*, abs/1906.02629.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. [fairseq: A fast, extensible toolkit for sequence modeling](#). In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. 2018. [Scaling neural machine translation](#). In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 1–9, Brussels, Belgium. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. [Bleu: a method for automatic evaluation of machine translation](#). In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Gabriel Pereyra, George Tucker, Jan Chorowski, Lukasz Kaiser, and Geoffrey E. Hinton. 2017. [Regularizing neural networks by penalizing confident output distributions](#). In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*.

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *ArXiv*, abs/1706.05098.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. 2016. [Improved techniques for training gans](#). In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 2234–2242. Curran Associates, Inc.

- Holger Schwenk. 2007. *Continuous space language models*. *Comput. Speech Lang.*, 21(3):492–518.
- Holger Schwenk, Daniel Dechelotte, and Jean-Luc Gauvain. 2006. *Continuous space language models for statistical machine translation*. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 723–730, Sydney, Australia. Association for Computational Linguistics.
- Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. *Neural machine translation of rare words with subword units*. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.
- Claude E. Shannon. 1948. *A mathematical theory of communication*. *The Bell System Technical Journal*, 27:379–423, 623–656.
- Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. *Dropout: A simple way to prevent neural networks from overfitting*. *Journal of Machine Learning Research*, 15:1929–1958.
- Felix Stahlberg and Bill Byrne. 2019. *On NMT search errors and model errors: Cat got your tongue?* In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3356–3362, Hong Kong, China. Association for Computational Linguistics.
- Martin Sundermeyer, Ralf Schütter, and Hermann Ney. 2012. Lstm neural networks for language modeling. In *Interspeech*, pages 194–197, Portland, OR, USA.
- Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014a. *Sequence to sequence learning with neural networks*. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc.
- Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014b. *Sequence to sequence learning with neural networks*. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS’14, pages 3104–3112, Cambridge, MA, USA. MIT Press.
- C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. 2016. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826.
- C. Szegedy, Wei Liu, Yangqing Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. 2015. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. *Attention is all you need*. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.
- Jinzhuo Wang, Wenmin Wang, and Wen Gao. 2018. *Beyond knowledge distillation: Collaborative learning for bidirectional model assistance*. *IEEE Access*, 6:39490–39500.
- Shuo Wang, Yang Liu, Chao Wang, Huanbo Luan, and Maosong Sun. 2019. *Improving back-translation with uncertainty-based confidence estimation*. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 791–802, Hong Kong, China. Association for Computational Linguistics.
- Lingxi Xie, Jingdong Wang, Zhen Wei, Meng Wang, and Qi Tian. 2016. *Disturblabel: Regularizing cnn on the loss layer*. In *CVPR*, pages 4753–4762.
- Long Zhou, Jiajun Zhang, and Chengqing Zong. 2019. *Synchronous bidirectional neural machine translation*. *Transactions of the Association for Computational Linguistics*, 7:91–105.
- Zhiming Zhou, Han Cai, Shu Rong, Yuxuan Song, Kan Ren, Weinan Zhang, Jun Wang, and Yong Yu. 2017. Activation maximization generative adversarial nets. In *ICLR*.

## A Derivation of Optimal Solutions

Ignoring the outer summation in tokens and dropping the dependencies on  $n$  for simplicity, the optimization problem in Equation 8 can be solved analytically and the optimization problem in Equation 10 can be solved numerically.

### A.1 Minimizing $L_n^{\text{STD}}$

$L_n^{\text{STD}}$  takes the form of  $\sum_x p \log q$ , where both  $p$  and  $q$  are probability distributions in  $x$ . The divergence inequality can be directly applied:

$$\begin{aligned} L_n^{\text{STD}} &= \sum_{v=1}^V - \left( (1-m)p_v + m\frac{1}{V} \right) \log q_v \\ &\geq \sum_{v=1}^V - \left( (1-m)p_v + m\frac{1}{V} \right) \cdot \\ &\quad \log \left( (1-m)p_v + m\frac{1}{V} \right) \end{aligned}$$

Alternatively, one can use the Lagrange multiplier and calculate first order derivatives:

$$\begin{aligned}\mathcal{L}_n^{\text{STD}}(q_v, \lambda) &= L_n^{\text{STD}} + \lambda \left( \sum_v q_v - 1 \right) \\ \frac{\partial \mathcal{L}_n^{\text{STD}}}{\partial q_v} &= \frac{(1-m)p_v + m\frac{1}{V}}{q_v} + \lambda \\ \frac{\partial \mathcal{L}_n^{\text{STD}}}{\partial \lambda} &= \sum_v q_v - 1\end{aligned}$$

Afterwards, set them to zero and solve for  $\lambda$ :

$$\begin{aligned}\frac{\partial \mathcal{L}_n^{\text{STD}}}{\partial q_v} = 0 &\Rightarrow q_v = \frac{(1-m)p_v + m\frac{1}{V}}{-\lambda} \\ \frac{\partial \mathcal{L}_n^{\text{STD}}}{\partial \lambda} = 0 &\Rightarrow \lambda = -1\end{aligned}$$

Plugging  $\lambda$  back in yield  $q_v$ , which should be further checked to see if it is a maxima or minima.

In both methods, the minimum is obtained when:

$$\tilde{q}_v^{\text{STD}} = (1-m)p_v + m\frac{1}{V}$$

## A.2 Minimizing $L_n^{\text{CFD}}$

Applying the Lagrange multiplier, the first order derivatives can be derived:

$$\begin{aligned}\mathcal{L}_n^{\text{CFD}}(q_v, \lambda) &= L_n^{\text{CFD}} + \lambda \left( \sum_v q_v - 1 \right) \\ \frac{\partial \mathcal{L}_n^{\text{CFD}}}{\partial q_v} &= \frac{-(p_v - m'q_v)}{q_v} + m' \log q_v + \lambda \\ \frac{\partial \mathcal{L}_n^{\text{CFD}}}{\partial \lambda} &= \sum_v q_v - 1\end{aligned}$$

Note that setting  $\frac{\partial \mathcal{L}_n^{\text{CFD}}}{\partial q_v}$  to zero results in a transcendental equation in the form of:

$$Ax + bx \log x = C$$

where  $A = (m' + \lambda)$ ,  $B = m'$ ,  $C = p_v$  and  $x = q_v$ .

Consider that the Lambert  $W$  function is the inverse function of:

$$f(W) = We^W$$

we can rewrite the transcendental equation until we

reach a similar form:

$$\begin{aligned}Ax + Bx \log x &= C \\ \xrightarrow{t=\frac{1}{x}} \frac{A}{t} - \frac{B \log t}{t} &= C \\ A &= Ct + B \log t \\ \frac{A}{B} &= \frac{C}{B}t + \log t \\ \xrightarrow{u=\frac{C}{B}t} \frac{A}{B} &= u + \log u + \log \frac{B}{C} \\ u + \log u &= \frac{A}{B} + \log \frac{C}{B} \\ ue^u &= \frac{C}{B}e^{\frac{A}{B}} \\ \Rightarrow u &= W\left(\frac{C}{B}e^{\frac{A}{B}}\right)\end{aligned}$$

reversing the variable replacements:

$$\begin{aligned}u &= \frac{C}{B}t = \frac{C}{B} \cdot \frac{1}{x} \\ \Rightarrow x &= \frac{C}{B} \cdot \frac{1}{u} \\ &= \frac{C}{BW\left(\frac{C}{B}e^{\frac{A}{B}}\right)}\end{aligned}$$

Finally, plugging in  $A$ ,  $B$  and  $C$ , we arrive at Equation 11:

$$\tilde{q}_v^{\text{CFD}} = \frac{p_v}{m'W_0\left(\frac{p_v}{m'}e^{1+\frac{\lambda}{m'}}\right)}$$

When  $p$  is a one hot distribution and  $m'$  is given, one can use the constraint of  $q_v$  being a probability distribution to numerically solve for  $\lambda$ . Once  $\lambda$  is obtained, actual values of  $\tilde{q}_v^{\text{CFD}}$  can be calculated.

# Comparing Probabilistic, Distributional and Transformer-Based Models on Logical Metonymy Interpretation

Giulia Rambelli

University of Pisa

giulia.rambelli@phd.unipi.it

Emmanuele Chersoni

Hong Kong Polytechnic University

emmanuelechersoni@gmail.com

Alessandro Lenci

University of Pisa

alessandro.lenci@unipi.it

Philippe Blache

Aix-Marseille University Hong Kong Polytechnic University

Chu-Ren Huang

Hong Kong Polytechnic University

## Abstract

In linguistics and cognitive science, *Logical metonymies* are defined as type clashes between an event-selecting verb and an entity-denoting noun (e.g. *The editor finished the article*), which are typically interpreted by inferring a hidden event (e.g. *reading*) on the basis of contextual cues.

This paper tackles the problem of logical metonymy *interpretation*, that is, the retrieval of the covert event via computational methods. We compare different types of models, including the probabilistic and the distributional ones previously introduced in the literature on the topic. For the first time, we also tested on this task some of the recent Transformer-based models, such as BERT, RoBERTa, XLNet, and GPT-2.

Our results show a complex scenario, in which the best Transformer-based models and some traditional distributional models perform very similarly. However, the low performance on some of the testing datasets suggests that logical metonymy is still a challenging phenomenon for computational modeling.

## 1 Introduction

The phenomenon of *logical metonymy* is defined as a type clash between an event-selecting metonymic verb (e.g., *begin*) and an entity-denoting nominal object (e.g., *the book*), which triggers the recovery of a hidden event (e.g., *reading*). Logical metonymies have been widely studied, on the one hand, in theoretical linguistics as they represent a challenge to traditional theories of compositionality (Asher, 2015; Pustejovsky and Batiukova, 2019). On the other hand, they received extensive attention in cognitive research on human sentence processing as they determine extra processing costs during online sentence comprehension (McElree et al., 2001; Traxler et al., 2002), apparently related

to “the deployment of operations to construct a semantic representation of the event” (Frisson and McElree, 2008).<sup>1</sup>

Logical metonymy has also been explained in terms of the *words-as-cues hypothesis* proposed by Jeffrey Elman (Elman, 2009, 2014). This hypothesis relies on the experimental evidence that human semantic memory stores knowledge about events and their typical participants (see McRae and Matsuki (2009) for an overview) and claims that words act like cues to access event knowledge, incrementally modulating sentence comprehension. The results obtained in a probe recognition experiment by Zarcone et al. (2014), in line with this explanation, suggest that speakers interpret logical metonymies by inferring the most likely event the sentences could refer to, given the contextual cues. Previous research in NLP on logical metonymy has often been influenced by such theoretical explanation (Zarcone and Padó, 2011; Zarcone et al., 2012; Chersoni et al., 2017).

In our contribution, we propose a general comparison of different classes of computational models for logical metonymy. To begin with, we tested two approaches that have been previously introduced in the literature on the topic: probabilistic and distributional models (Zarcone et al., 2012). We also examined the Structured Distributional Model (SDM) by Chersoni et al. (2019), which represents sentence meaning with a combination of formal structures and distributional embeddings to dynamically integrate knowledge about events and their typical participants, as they are activated by lexical items. Finally, to the best of our knowledge, we are the first ones to include the recent Transformer language models into a contrastive study on

<sup>1</sup>Notice however that the evidence is not uncontroversial: Delogu et al. (2017) report that coercion costs largely reflect word surprisal, without any specific effect of type shift in the early processing measures.

logical metonymy. Transformers (Vaswani et al., 2017; Devlin et al., 2019) are the dominant class of NLP systems in the last few years, since they are able to generate “dynamic” representations for a target word depending on the sentence context. As the interpretation of logical metonymy is highly sensitive to context, we deem that the contextual representations built by Transformers might be able to integrate the covert event that is missing in the surface form of the sentence.

All models are evaluated on their capability of **assigning the correct interpretation to a metonymic sentence**, that is, **recovering the verb that refers to the correct interpretation**. This task is hard for computational models, as they must exploit contextual cues to distinguish covert events with a high typicality (e.g., *The pianist begins the symphony* → *playing*) from plausible but less typical ones (→ *composing*).

## 2 Related Work

### 2.1 Computational Models of Logical Metonymy

According to Zarcone et al. (2013), the phenomenon of logical metonymy can be explained in terms of the *thematic fit*, that is, the degree of compatibility between the verb and one of its arguments (the direct object, in this case). On the one hand, a low thematic fit between an event-selecting verb and an entity-denoting argument triggers the recovery of a covert event, while on the other hand, the recovered event is often the best fitting one, given the information available in the sentence.

Research in NLP on logical metonymy initially focused on the problem of covert event retrieval, which was tackled by means of probabilistic models (Lapata and Lascarides, 2003; Shutova, 2009), or by using Distributional Semantic Models (DSMs) that identify the candidate covert event with the one that has the highest thematic fit with the arguments in the sentence (Zarcone et al., 2012). Following the psycholinguistic works by McElree et al. (2001) and Traxler et al. (2002), which reported increased reading times and longer fixations in eye-tracking for the metonymic sentences, Zarcone et al. (2013) proposed a distributional model of the thematic fit between verb and object, and showed that it accurately reproduces the differences between the experimental conditions in the data from the two original studies.

A general distributional model for sentence com-

prehension was used by Chersoni et al. (2017) to simultaneously tackle both these two aspects of logical metonymy (covert event retrieval and increased processing times), although at the cost of a highly-elaborated compositional model. The authors recently introduced a more up-to-date and refined version of their sentence comprehension model (Chersoni et al., 2019), but it has not been tested on the logical metonymy task so far.

### 2.2 Transformer Models in NLP

The traditional approach in Distributional Semantics has been the building of a single, stable vector representation for each word type in the corpus (Turney and Pantel, 2010; Lenci, 2018). Lately, a new generation of embeddings has emerged, in which each occurrence of a word in a specific sentence context gets a unique representation (Peters et al., 2018). The most recent systems typically rely on an LSTM or a Transformer architecture for getting word representations: they are trained on large amounts of textual data and the word vectors are learned as a function of the internal states of the encoder, such that a word in different sentence contexts determines different activation states and is represented by a different vector. Thus, embeddings generated by these new models are said to be *contextualized*, as opposed to the *static* vectors generated by the earlier frameworks, and they aim at modeling the specific sense assumed by the word in context. One of the most popular and successful contextualized model is probably BERT (Devlin et al., 2019), whose key technical innovation is applying the bidirectional training of Transformer, a popular attention model, to language modelling. This is in contrast to previous efforts which looked at a text sequence either from left to right or combined left-to-right and right-to-left training. The results of the paper show that a language model with bidirectional training can have a deeper sense of language context and structure than single-direction language models.

An interesting aspect of Transformer models like BERT is that they are trained via **masked language modeling**, that is, they have to retrieve a word that has been masked in a given input sentence. Since interpreting logical metonymy implies the retrieval of an event that is not overtly expressed and that humans retrieve integrating the lexical cues in the sentence, these models are potentially a very good fit for this task. To draw an analogy, we could

imagine that the covert event is a verb that has been ‘masked’ in the linguistic input and that we ask BERT-like models to make a guess.

It is important to point out that not all Transformers are used for masked language modeling: among those tested for this study, BERT and RoBERTa are directly trained with this objective, XLNet is trained with permutation language modeling, but can still retrieve a hidden word given a bidirectional context, and GPT-2 works similarly to a traditional, unidirectional language model.

### 3 Experimental Settings

#### 3.1 Task

Our research question focuses on how computational models can interpret metonymic sentences. To explore this issue, we define the task of logical metonymy interpretation as a **covert event recovery** task. More specifically, given a sentence like *The architect finished the house*, the computational model has to return the most likely hidden verb for the sentence, i.e. the covert event representing its interpretation. Despite the architectural differences, all tested models compute a *plausibility score* of a verb as expressing the covert event associated with a <subject, metonymic verb, object> triple. We evaluate the scores returned by a model against human judgments using the standard measures of accuracy and correlation depending if the dataset contains categorical or continuous variables.

#### 3.2 Datasets

In our experiments, we use three datasets designed for previous psycholinguistic studies, and a newly created one by means of an elicitation task.

The **McElree** dataset (**MC**) comprises the stimuli from the sentences of the self-paced reading experiment of [McElree et al. \(2001\)](#) and includes 30 pairs of tuples. Each pair has the same subject, metonymic verb, object, just the covert verb varies. As in the conditions of the original experiment, the hidden verb could be either highly plausible, or plausible but less typical, given the subject and the object of the tuple. The **Traxler** dataset (**TR**) results from the sentences of the eye-tracking experiment of [Traxler et al. \(2002\)](#) and includes 36 pairs of tuples. The format is the same as the McElree dataset. On these two datasets, the models have to perform a *binary classification task*, with the goal of assigning a higher score to the covert event in the typical condition.

The **Lapata-Lascarides** dataset (**L&L**) ([Lapata and Lascarides, 2003](#)) includes 174 tuples, each composed by a metonymic verb, an object and a potential covert verb. The authors collected plausibility ratings for each metonymy by turning the tuples into sentences and used the Magnitude Estimation Paradigm ([Stevens, 1957](#)) to ask human subjects to rate the plausibility of the interpretation of the metonymic verb. Finally, the mean ratings have been normalized and log-transformed.

A further dataset of **recovered covert events** (**CE**) was collected by the authors. The metonymic sentences used in the McElree and Traxler experiments were turned into 69 templates with an empty slot corresponding to the covert event (e.g., *The student began \_\_\_ the book late in the semester*). Thirty subjects recruited with crowdsourcing were asked to produce two verbs that provided the most likely fillers for the event slot. Out of the 4,084 collected verbs, we selected those with a production frequency  $\geq 3$  for a given stimulus. The final dataset comprises 285 items each consisting of a subject – metonymic verb – object tuple  $t$  and a covert event  $e$  associated with a salience score corresponding to the event conditional probability given the tuple  $P(e|t)$  (i.e., the production frequency of  $e$  normalized by the total events produced for  $t$ ). In the case of the latter two datasets, for each model we compute the Spearman’s correlation between the probabilities generated by the model and the human judgements. Examples from these datasets are provided in Table 1.

While collecting the data for **CE**, we also run a statistical comparison between the production frequencies of the verbs in the typical and in the atypical condition that appear in the binary classification datasets, to ensure that humans genuinely agree on the higher typicality of the former. The result confirmed this assumption: according to the Wilcoxon signed rank test with continuity correction, the frequencies of production of the typical verbs for the **MC** dataset were significantly higher ( $W = 424, p < 0.001$ ), and the same holds for the typical verbs in the **TR** dataset ( $W = 526.5, p < 0.001$ ).

#### 3.3 Models

In the following section, we describe the general aspects of the computational models that we tested on logical metonymy interpretation.

Dataset	Subject-verb-object	Covert event	Condition/Score	Size
<b>MC</b>	chef start dinner	<i>prepare</i> <i>eat</i>	HIGH_TYP LOW_TYP	30 (pairs)
<b>TR</b>	dieter resist cake	<i>eat</i> <i>taste</i>	HIGH_TYP LOW_TYP	36 (pairs)
<b>L&amp;L</b>	— start experiment	<i>implement</i> <i>study</i>	0.1744 0.0184	174
<b>CE</b>	architect start house	<i>draw</i> <i>build</i>	0.348 0.087	258

Table 1: Examples of stimuli from each dataset.

### 3.3.1 Probabilistic Model

As a baseline model, we adopt the simple probabilistic approach proposed by [Lapata and Lascarides \(2003\)](#) and replicated by [Zarcone et al. \(2012\)](#) as the **SO<sub>p</sub>** model, which was reported as the best performing probabilistic model on the task. The interpretation of a logical metonymy (e.g., *The pianist began the symphony*) is modelled as the joint distribution  $P(s, v, o, e)$  of the variables  $s$  (the subject, *pianist*),  $v$  (the metonymic verb, *began*),  $o$  (the object, *symphony*), and the covert event  $e$  (e.g., *play*). We compute that probability considering the metonymic verb constant:

$$P(s, v, o, e) \approx P(e)P(o|e)P(s|e)$$

The verb  $E$  representing the preferred interpretation of the metonymy is the verb  $e$  maximizing the following equation:

$$E = \text{argmax}_e P(e)P(o|e)P(s|e)$$

We computed the statistics from a 2018 dump of the English Wikipedia, parsed with the Stanford CoreNLP toolkit ([Manning et al., 2014](#)).

Dataset	Coverage
<b>MC</b>	19/30 (pairs)
<b>TR</b>	21/36 (pairs)
<b>L&amp;L</b>	151/174 (items)
<b>CE</b>	195/285 (items)

Table 2: Coverage for the probabilistic model.

### 3.3.2 Logical Metonymy as Thematic Fit

Distributional models of logical metonymy assume that the event recovery task can be seen as a *thematic fit* task: recovering the covert event means identifying the verb with the highest thematic fit with the metonymic sentence. We reimplement the

distributional model by [Zarcone et al. \(2012\)](#) with the following procedure:

- we retrieve the  $n$  ( $= 500$ )<sup>2</sup> most strongly associated verbs for the subject and the object respectively, and we take the intersection of the two lists;
- we update their association scores using either the sum (*add*) or the product (*prod*) function;
- we select the embeddings corresponding to the first  $m$  ( $= 20$ ) verbs in this list and we add them together to create the prototype vector of the verb given the subject and the object;
- the thematic fit of the covert event  $e$  with respect to the nominal entities is computed as the similarity score of its corresponding lexical vector  $\vec{e}$  with the prototype vector. As we did the probabilistic model, we discard the metonymic verb from this computation.<sup>3</sup>

We test two variations of this model, **TF-add** and **TF-prod**, which differ for the filler selection update function. Statistics were extracted from Wikipedia 2018, and the vectors were the publicly-available Wikipedia embeddings<sup>4</sup> trained with the FastText model ([Bojanowski et al., 2017](#)). The verb-filler association score is the Local Mutual Information ([Evert, 2008](#)). Similarly, the scores for the subject fillers are defined as:

$$LMI(s, e) = f(e \xleftarrow{\textit{sbj}} s) \log_2 \frac{p(s|e)}{p(s)p(e)}$$

<sup>2</sup>We set a high value for this parameter in order to maximize the coverage.

<sup>3</sup>[Zarcone et al. \(2012\)](#) show that, for both the probabilistic and the distributional model, including the metonymic verb does not help too much in terms of performance and leads to coverage issues.

<sup>4</sup><https://fasttext.cc/docs/en/english-vectors.html>

where  $s$  is the subject,  $e$  the covert event, and  $f(e \xleftarrow{obj} s)$  indicates the frequency of  $e$  with the subject. The scores for the object position are computed with the following formula:

$$LMI(o, e) = f(e \xleftarrow{obj} o) \log_2 \frac{p(o|e)}{p(o)p(e)}$$

where  $o$  is the object and  $f(e \xleftarrow{obj} o)$  represents the joint frequency of  $e$  with the object.

### 3.3.3 Structured Distributional Model

The **Structured Distributional Model** (SDM) proposed by Chersoni et al. (2019) consists of two components: a *Distributional Event Graph* (henceforth, *DEG*), and a meaning composition function. *DEG* represents event knowledge as a graph automatically built from parsed corpora, where the nodes are words associated to a numeric vector, and the edges are labeled with syntactic relations and weighted using statistic association measures. Each event is represented as a path in *DEG*, that is, a sequence of edges (relations) which joins a sequence of vertices (words). Thus, given a lexical cue  $w$ , it is possible to identify the associated events and to generate expectations about incoming inputs on both the paradigmatic and the syntagmatic axis.

The composition function makes use of two semantic structures (inspired by DRT (Kamp, 2013)): the *linguistic condition* ( $LC$ ), a context-independent tier of meaning, and the *active context* ( $AC$ ), which accumulates contextual information available during sentence processing or activated by lexical items. The crucial aspect is that the model associates a vectorial representation to these formal structures:  $\vec{LC}$  is the sum of the embeddings of the lexical items of a sentence;  $\vec{AC}$ , for each syntactic slot, is represented as the centroid vector built out of the role vectors  $\vec{r}_1, \dots, \vec{r}_n$  available in  $AC$ , i.e. the syntactic associates of the lexical items that have been already processed.

In our implementation of SDM, the *DEG* is constructed by extracting syntactic relations from the same dump of Wikipedia adopted in the previous models, and we chose as lexical embeddings the same FastText Wikipedia vectors. Following the same assumption of the previous experiment, we model the covert event recovery task as a thematic fit task: the goal is to predict the hidden verb on the basis of the subject and the object, treating the metonymic verb as a constant. Specifically, the model builds a semantic representation for each

	Model settings				Data size
	L	H	A	P	
<b>BERT</b> large-cased	24	1024	16	340M	16GB
<b>RoBERTa</b> large	24	1024	16	355M	160GB
<b>XLNet</b> large-cased	24	1024	16	340M	113GB
<b>GPT-2</b> extra-large	48	1600	25	1542M	40 GB

Table 3: Comparison between transformer models. Model details:  $L$ : number of layers,  $H$ : dimension of hidden states,  $A$ : attention head numbers, and  $P$ : total parameter size.

tuple in the dataset. The linguistic condition vector  $\vec{LC}$  contains the sum of the subject and object embeddings. At the same time, the event knowledge vector  $\vec{AC}$  contains the prototypical embedding for the main verb, using *DEG* to retrieve the most associated verbs for the subject and the object, as in Chersoni et al. (2019). The scoring function has been adapted to the event recovery task as follows:

$$\cos(\vec{e}, \vec{LC}(sent)) + \cos(\vec{e}, \vec{AC}(sent))$$

where  $sent$  refers to the metonymic test tuple. In other words, we quantify the typicality of a verb for a tuple subject-object as the sum of i.) the cosine similarity between the event embedding and the additive combination of the other argument vectors ( $\vec{LC}$ ) and ii.) the cosine similarity between the event embedding and the prototype vector representing the active context ( $\vec{AC}$ ).

### 3.3.4 Transformer-based Models

We experiment with four Transformer models which have been shown to obtain state-of-the-art performances on several NLP benchmarks.

The popular **BERT** model (Devlin et al., 2019) was the first to adopt the bidirectional training of Transformer for a language modeling task. To make this kind of training possible, BERT introduced a masked language modeling objective function: random words in the input sentences are replaced by a [MASK] token and the model attempts to predict the masked token based on the surrounding context. Simultaneously, BERT is optimized on a next sentence prediction task, as the model receives sentence pairs in input and has to predict whether the second sentence is subsequent to the

first one in the training data.<sup>5</sup> BERT has been trained on a concatenation of the BookCorpus and the English Wikipedia, for a total of 3300M tokens ca. In our experiments, we used the larger pre-trained version, called BERT-large-cased.

**RoBERTa** (Liu et al., 2019) has the same architecture as BERT, but it introduces several parameter optimization choices: it makes use of dynamic masking (compared to the static masking of the original model), of a larger batch-size and a larger vocabulary size. Moreover, the input consists of complete sentences randomly extracted from one or multiple documents, and the next sentence prediction objective is removed. Besides the optimized design choice, another key difference of RoBERTa with the other models is the larger training corpus, which consists of a concatenation of the BookCorpus, CCNEWS, OpenWebText, and STORIES. With a total 160GB of text, RoBERTa has access to more potential knowledge than the other models. For our tests, we used the large pre-trained model.

**XLNet** (Yang et al., 2019) is a generalized autoregressive (AR) pretraining method which uses the context words to predict the next word. The AR architecture is constrained to a single direction (either forward or backwards), that is, context representation takes in consideration only the tokens to the left or to the right of the  $i$ -th position, while BERT representation has access to the contextual information on both sides. To capture bidirectional contexts, XLNet is trained with a permutation method as language modeling objective, where all tokens are predicted but in random order. XLNet’s training corpora were the same as BERT plus Giga5, ClueWeb 2012-B and Common Crawl, for a total of 32.89B subword piece. Also in this case, we used the large pre-trained model.

**GPT-2** (Radford et al., 2019), a variation of GPT, is a uni-directional transformer language model, which means that the training objective is to predict the next word, given all of the previous words. Compared with GPT, GPT-2 optimizes the layer normalization, expands the vocabulary size to 50,257, increases the context size from 512 to 1024 tokens, and optimizes with a larger batch size of 512. In addition, GPT-2 is pre-trained on WebText, which was created by scraping web pages, for a total of 8 million documents of data (40 GB). We

---

<sup>5</sup>Notice that the usefulness of this secondary objective function was questioned, and it was indeed removed in more recent models (Yang et al., 2019; Liu et al., 2019; Joshi et al., 2020).

used the XL version of GPT-2 for our experiments.

The parameters of the Transformer models are reported in Table 3. BERT, RoBERTa and XLNet are used to perform a *word prediction task*: given a sentence and a masked word in position  $k$ , they compute the probability of a word  $w_k$  given the  $context_k$ :  $P(w_k | context_k)$ . For our experiments, the context is the entire sentence  $S$  with the  $k$ -th word (the covert event) being replaced by a special token ‘[MASK]’. Therefore, we turned the test tuples into full sentences, masking the verb as in the example below: *The architect finishes [MASK] house.*<sup>6</sup> We then compute the probability of a hidden verb to occur in that position, and we expect the preferred verb to get a high value. We performed this task using the packages of the HappyTransformer library.<sup>7</sup>

As GPT-2 works as a traditional language model, we adopted this model to calculate the probability of the entire sentence (instead of the probability of the hidden verb given the context). In this case, we expect that sentences evoking more typical events get higher values. We adopted the lm-scorer package to compute sentence probabilities.<sup>8</sup>

## 4 Evaluation Results

Table 5 and 4 report the final evaluation scores. The performance of the probabilistic model is in line with previous studies, and it outperforms distributional models in some cases, proving that it is indeed a hard baseline to beat. However, accuracy and correlation are computed only on a subgroup of the test items: actually, the model covers about 60% of the datasets’ tuples (86.8% for L&L), as we reported in Table 2. Coverage is the main issue probabilistic models have to face (Zarcone et al., 2013), while distributional models do not experience such limitation.

Regarding the thematic fit models, we observe that there is no difference between the TF-add and TF-prod models, as they obtain similar scores.

---

<sup>6</sup>One of the anonymous reviewers argues that the performance of the Transformer-based models might be influenced by the prompt sentence and suggest more variations of the input sentences. We indeed tested several manipulations of the inputs before feeding them to the transformers, changing 1) the tense of the metonymic verb (using the past tense) and 2) the number of the direct object (we used the plurals of the dataset nouns). However, the results did not show any consistent trend.

<sup>7</sup><https://github.com/EricFillion/happy-transformer>

<sup>8</sup><https://pypi.org/project/lm-scorer/>

	Probabilistic	Distributional			Transformer-based			
		<i>SOp</i>	<i>TF-add</i>	<i>TF-prod</i>	<i>SDM</i>	<i>BERT</i>	<i>RoBERTa</i>	<i>XLNet</i>
<b>MC</b>	0.68		0.70	0.73	0.77	0.70	0.80	0.40
<b>TR</b>	0.48		0.53	0.53	<b>0.72</b>	0.47	<b>0.72</b>	0.39
<b>O. P.</b>	0.58		0.62	0.63	0.75	0.59	0.76	0.40

Table 4: Results for binary classification task.

	Probabilistic	Distributional			Transformer-based			
		<i>SOp</i>	<i>TF-add</i>	<i>TF-prod</i>	<i>SDM</i>	<i>BERT</i>	<i>RoBERTa</i>	<i>XLNet</i>
<b>L&amp;L</b>	0.53		0.41	0.41	0.53	0.61	<b>0.73</b>	0.04
<b>CE</b>	0.36		0.26	0.22	<b>0.40</b>	0.27	0.39	0.18
<b>O. P.</b>	0.45		0.34	0.32	0.47	0.44	<b>0.56</b>	0.11

Table 5: Results for correlation task.

However, we need to point out that, when the system computes the intersection of the two lists of the top verbs for subjects and objects, sometimes the number of retrieved items is less than 20 (the model parameter for the verb embedding selection, cf. Section 3.3.2). Therefore, independently of the selected function, the verbs used to compute the prototypical vector are eventually all those belonging to the intersection. Moreover, TF-models are often close to, and never significantly outperform the probabilistic baseline.

Among the distributional models, SDM is the one that obtains a considerable performance across all the datasets. This model performs close to RoBERTa both in the Traxler and in the CE dataset. This result is surprising, considering that SDM is trained just on a dump of Wikipedia, while RoBERTa is trained on 160 GB of text and implements advanced deep learning techniques. This outcome confirms that SDM, which has been designed to represent event knowledge and the dynamic construction of sentence meaning, is able to adequately model the typicality of events. This aspect has been suggested to be one of the core components of the language processing system (Baggio and Hagoort, 2011; Baggio et al., 2012; Chersoni et al., 2019).

On the other hand, Transformers also provided interesting results. RoBERTa achieves the best score for the L&L dataset, reaching a statistical significance of the improvement over SDM ( $p < 0.01$ ).<sup>9</sup> More importantly, it is the only Transformer that consistently obtains good results across all datasets, while the scores from other

Transformer models are highly fluctuating. We believe that the gigantic size of the training corpus is a factor that positively affects its performance. At the same time, GPT-2 achieves the highest score for MC dataset (0.87) (but the improvement over RoBERTa and SDM does not reach statistical significance), although it performs significantly lower on the other benchmarks<sup>10</sup>.

For the sake of completeness, we also report the overall performance of each model over the two tasks. Results identify RoBERTa and GPT-2 as the best models for the correlation and classification tasks, respectively. However, we wonder if the average score is a valid measure to identify the best model. These two models tend to have a wavering behavior, which results in large differences between the two datasets scores. Specifically, Roberta achieves 0.75 for the L&L dataset, but only 0.39 for the CE one, with 0.36 points of difference. Similarly, GPT-2 reaches 0.89 scores for the MC dataset, but its performance goes down by 0.16. On the contrary, SDM behavior is more stable, with a smaller gap between the two datasets’ scores (0.13 point difference for the correlation task and just 0.05 for the accuracy task).

#### 4.1 Error analysis

**Binary classification task** For the MC and TR datasets, we evaluate the models for their capability of assigning a higher probability to the verb in the typical condition. It is important to emph-

<sup>9</sup>The  $p$ -value is computed with Fisher’s r-to-z transformation, one-tailed test.

<sup>10</sup>We determine the significance of differences between models for MC and TR datasets with a McNemar’s Chi-Square Test, applied to a 2x2 contingency matrix containing the number of correct and incorrect answers (replicating the approach of Zarcone et al. (2012)).

size that both verbs are plausible in the context, but one describes a more likely event given the subject and the object. This remark is essential, because it explains the performance of all models, distributional and Transformer ones.

To identify which tuples are the most difficult ones, we built a heat map visualizing the correctly-predicted ones in blue, and the wrong ones in yellow (see Figures 1 and 2). We do not consider the accuracy values obtained by the probabilistic model for its partial coverage.

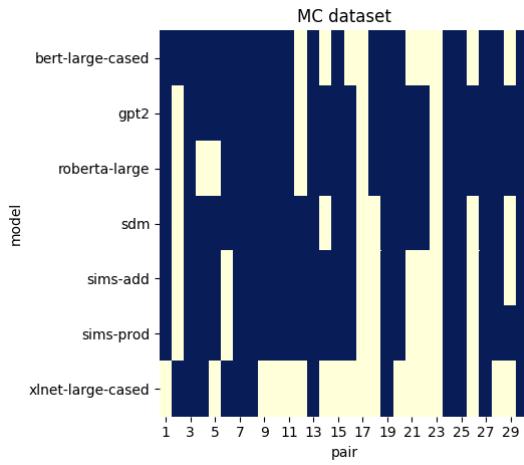


Figure 1: Heat map for error analysis over MC dataset.

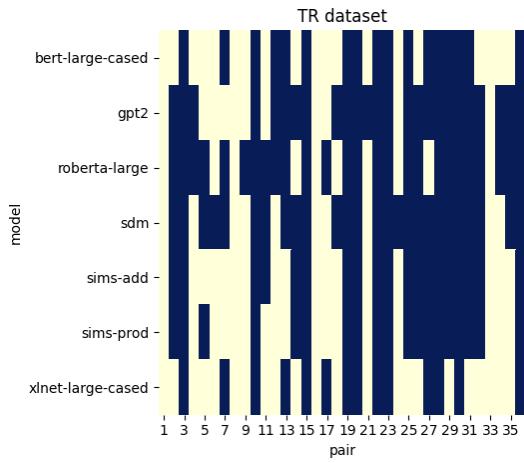


Figure 2: Heat map for error analysis over TR dataset.

This visualization technique reveals that some pairs are never predicted correctly, corresponding to the fully vertical yellow lines in the figures. In what follows we report the tuples that are consistently mistaken for MC (1) and TR (2) datasets.

- (1) a. *The teenager starts the novel.*  
b. *The worker begins the memo.*

- (2) a. *The editor finishes the newspaper.*  
b. *The director starts the script.*  
c. *The teenager begins the novel.*

In all the above cases, a model must discriminate between the verb *read* (HIGH\_TYP) and *write* (LOW\_TYP).<sup>11</sup> It is interesting to notice that, for many of the *read-write* pairs in the binary classification data, the production frequencies of typical and atypical verb are much closer than on average, suggesting that the interpretation requires understanding of subtle nuances of context-sensitive typicality, which might not be trivial even for humans.

Furthermore, in Figure 2 we observe that for two TR's pairs, SDM is the only one picking the right choice: *The stylist starts the braid* and *The auditor begins the taxes*. It seems that models regularly tend to prefer a verb with a more generic and undetermined meaning (*make* and *do*, respectively), while only SDM correctly assigns the HIGH\_TYP class to the verbs that indicate more precisely the manner of doing something (*braid* and *audit*).

On the other hand, GPT-2 and RoBERTa managed to pick the right choice for a few of the *read-write* items on which SDM is mistaken.

**Correlation task** Correlation is a more complex task compared to classification, as the lower scores also reveal. To better understand our results, we select the best model for the CE (i.e., SDM) and L&L (i.e., RoBERTa) datasets, and we plot the linear relationship between the human ratings and the model-derived probabilities.<sup>12</sup> For CE, Figure 3 reveals 1) a small positive correlation between the two variables, 2) a large amount of variance, and 3) a few outliers.

As for L&L in Figure 4, the majority of the points follow a roughly linear relationship, and there is a small variation around the trend. Nevertheless, this result could be influenced by the form of the input sentences. For all the other datasets, we masked the token between the verb and the object, and the corresponding hidden verb had to be in the progressive form (*The chef starts [cooking] dinner*). For L&L, instead, we chose to insert the preposition *to* after the verb since lots of the metonymic verbs (*want*, *try*, etc.) require to be followed by the infinitive verb. Thus, the context gives a higher

<sup>11</sup>Except for the sentence in 2.a, where the typical verb is *edit*.

<sup>12</sup>We apply the logarithmic transformation of data for visualization purposes.

probability to verbs as masked tokens, while different parts of speech could be equally plausible for the other conditions.

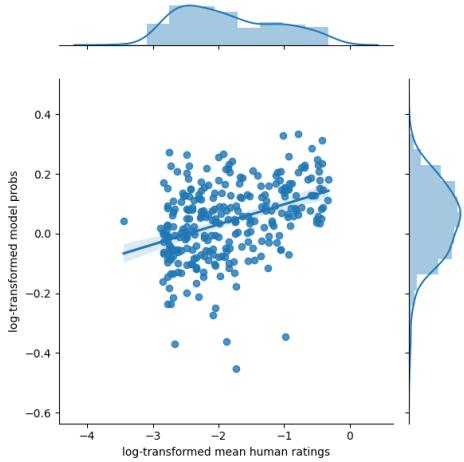


Figure 3: SDM correlation for CE.

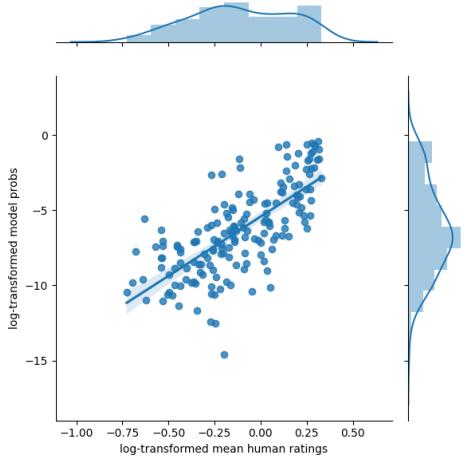


Figure 4: RoBERTa correlation for L&L.

## 5 Discussion and Conclusions

In this paper, we have presented a comparative evaluation of several computational models on the task of logical metonymy interpretation. We frame this problem as the retrieval of an event that is not overtly expressed in the surface form of the sentence. According to Elman’s Words-as-Cues framework, human subjects can infer the covert event in logical metonymy thanks to the generalized knowledge about events and participants stored in their semantic memory. Hence, during sentence processing, words in the sentence create a network of mutual expectations that triggers the retrieval of typical events associated with lexical items and gen-

erates expectations about the upcoming words (Elman, 2014). To tackle the task of logical metonymy interpretation, computational models must be able to recover unexpressed relationships between the words, using a context-sensitive representation of meaning that captures this event knowledge.

The most compelling outcome of the reported experiments is probably the performance of SDM, which achieves the best score for the TR and the CE datasets. These results demonstrate the significance of encoding event structures *outside* the embeddings (which are treated as nodes in a distributional graph), and the ability of the SDM compositional function to dynamically update the semantic representation for a sentence. However, the evaluation scores are not very high, especially in the correlation task. Results reveal that the contextualized information used by computational models is useful to recall plausible events connected to the arguments, but this is still not sufficient. Even Transformer models, which currently report state-of-the-art performances on several NLP benchmarks, are not performing significantly better than the SDM model, which is trained on a smaller corpus and without any advanced deep learning technique. Error analysis highlights that they are able to identify the plausible scenarios in which the participants could occur, but they still struggle in perceiving different nuances of typicality. Our experiments show how the logical metonymy task can be seen as a testing ground to check whether computational models encode common-sense event knowledge.

Future work might follow two directions. On the one hand, expanding the coverage of the graph could favourably increase the performance of SDM. On the other hand, Transformer models could be tested with new experimental settings, such as the fine-tuning of the pre-trained weights on thematic fit-related (Lenci, 2011; Sayeed et al., 2016; Santus et al., 2017) or semantic role classification tasks (Collobert et al., 2011; Zapirain et al., 2013; Roth and Lapata, 2015).

## 6 Acknowledgements

This work, carried out within the Institut Convergence ILCB (ANR-16-CONV-0002), has benefited from support from the French government, managed by the French National Agency for Research (ANR) and the Excellence Initiative of Aix-Marseille University (A\*MIDEX). We thank the anonymous reviewers for their insightful feedback.

## References

- Nicholas Asher. 2015. Types, Meanings and Coercions in Lexical Semantics. *Lingua*, 157:66–82.
- Giosuè Baggio and Peter Hagoort. 2011. The Balance between Memory and Unification in Semantics: A Dynamic Account of the N400. *Language and Cognitive Processes*, 26(9):1338–1367.
- Giosuè Baggio, Michiel Van Lambalgen, and Peter Hagoort. 2012. The Processing Consequences of Compositionality. *The Oxford Handbook of Compositionality*. Oxford, pages 657–674.
- Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics*, 5:135–146.
- Emmanuele Chersoni, Alessandro Lenci, and Philippe Blache. 2017. Logical Metonymy in a Distributional Model of Sentence Comprehension. In *Proceedings of \*SEM*.
- Emmanuele Chersoni, Enrico Santus, Ludovica Pannitto, Alessandro Lenci, Philippe Blache, and C-R Huang. 2019. A Structured Distributional Model of Sentence Meaning and Processing. *Natural Language Engineering*, 25(4):483–502.
- Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural Language Processing (Almost) from Scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537.
- Francesca Delogu, Matthew W Crocker, and Heiner Drenhaus. 2017. Teasing Apart Coercion and Surprise: Evidence from Eye-Movements and ERPs. *Cognition*, 161:46–59.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of NAACL-HLT 2019*, Minneapolis, MN.
- Jeffrey L Elman. 2009. On the Meaning of Words and Dinosaur Bones: Lexical Knowledge without a Lexicon. *Cognitive Science*, 33(4):547–582.
- Jeffrey L Elman. 2014. Systematicity in the Lexicon: On Having your Cake and Eating It Too. In Paco Calvo and John Symons, editors, *The Architecture of Cognition: Rethinking Fodor and Pylyshyn’s Systematicity Challenge*. The MIT Press, Cambridge, MA.
- Stefan Evert. 2008. Corpora and collocations. *Corpus linguistics. An international handbook*, 2:1212–1248.
- Steven Frisson and Brian McElree. 2008. Complement Coercion is not Modulated by Competition: Evidence from Eye Movements. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 34(1):1–11.
- Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving Pre-training by Representing and Predicting Spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.
- Hans Kamp. 2013. *Meaning and the Dynamics of Interpretation: Selected Papers by Hans Kamp*. Brill, Leiden-Boston.
- Mirella Lapata and Alex Lascarides. 2003. A Probabilistic Account of Logical Metonymy. *Computational Linguistics*, 29(2):261–315.
- Alessandro Lenci. 2011. Composing and Updating Verb Argument Expectations: A Distributional Semantic Model. In *Proceedings of the ACL Workshop on Cognitive Modeling and Computational Linguistics*.
- Alessandro Lenci. 2018. Distributional Models of Word Meaning. *Annual Review of Linguistics*, 4:151–171.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692*.
- Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland. Association for Computational Linguistics.
- Brian McElree, Matthew J Traxler, Martin J Pickering, Rachel E Seely, and Ray Jackendoff. 2001. Reading Time Evidence for Enriched Composition. *Cognition*, 78:B17–B25.
- Ken McRae and Kazunaga Matsuki. 2009. People Use their Knowledge of Common Events to Understand Language, and Do So as Quickly as Possible. *Language and Linguistics Compass*, 3(6):1417–1429.
- Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *Proceedings of NAACL*.
- James Pustejovsky and Olga Batiukova. 2019. *The Lexicon*. Cambridge University Press.
- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models Are Unsupervised Multitask Learners. *OpenAI Blog*, 1(8):9.
- Michael Roth and Mirella Lapata. 2015. Context-Aware Frame-Semantic Role Labeling. *Transactions of the Association for Computational Linguistics*, 3:449–460.

Enrico Santus, Emmanuele Chersoni, Alessandro Lenci, and Philippe Blache. 2017. Measuring Thematic Fit with Distributional Feature Overlap. In *Proceedings of EMNLP*.

Asad Sayeed, Clayton Greenberg, and Vera Demberg. 2016. Thematic Fit Evaluation: An Aspect of Selectional Preferences. In *Proceedings of the ACL Workshop on Evaluating Vector Space Representations for NLP*.

Ekaterina Shutova. 2009. Sense-Based Interpretation of Logical Metonymy Using a Statistical Method. In *Proceedings of the ACL-IJCNLP 2009 Student Research Workshop*, pages 1–9.

Stanley S Stevens. 1957. On the Psychophysical Law. *Psychological review*, 64(3):153.

Matthew J Traxler, Martin J Pickering, and Brian McElree. 2002. Coercion in Sentence Processing: Evidence from Eye-Movements and Self-Paced Reading. *Journal of Memory and Language*, 47(4):530–547.

Peter D Turney and Patrick Pantel. 2010. From Frequency to Meaning: Vector Space Models of Semantics. *Journal of Artificial Intelligence Research*, 37:141–188.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. *arXiv preprint arXiv:1906.08237*.

Benat Zapirain, Eneko Agirre, Lluis Marquez, and Mihai Surdeanu. 2013. Selectional Preferences for Semantic Role Classification. *Computational Linguistics*, 39(3):631–663.

Alessandra Zarcone, Alessandro Lenci, Sebastian Padó, and Jason Utt. 2013. Fitting, not Clashing! A Distributional Semantic Model of Logical Metonymy. In *Proceedings of IWCS*.

Alessandra Zarcone and Sebastian Padó. 2011. Generalized Event Knowledge in Logical Metonymy Resolution. In *Proceedings of CogSci*.

Alessandra Zarcone, Sebastian Padó, and Alessandro Lenci. 2014. Logical Metonymy Resolution in a Words-as-Cues Framework: Evidence from Self-paced Reading and Probe Recognition. *Cognitive Science*, 38(5):973–996.

Alessandra Zarcone, Jason Utt, and Sebastian Padó. 2012. Modeling Covert Event Retrieval in Logical Metonymy: Probabilistic and Distributional Accounts. In *Proceedings of the NAACL Workshop on Cognitive Modeling and Computational Linguistics*.

# AMR Quality Rating with a Lightweight CNN

Juri Opitz

Dept. of Computational Linguistics  
Heidelberg University  
69120 Heidelberg  
opitz@cl.uni-heidelberg.de

## Abstract

Structured semantic sentence representations such as Abstract Meaning Representations (AMRs) are potentially useful in various NLP tasks. However, the quality of automatic parses can vary greatly and jeopardizes their usefulness. This can be mitigated by models that can accurately rate AMR quality in the absence of costly gold data, allowing us to inform downstream systems about an incorporated parse’s trustworthiness or select among different candidate parses.

In this work, we propose to transfer the AMR graph to the domain of images. This allows us to create a simple convolutional neural network (CNN) that imitates a human judge tasked with rating graph quality. Our experiments show that the method can rate quality more accurately than strong baselines, in several quality dimensions. Moreover, the method proves to be efficient and reduces the incurred energy consumption.

## 1 Introduction

The goal of sentence meaning representations is to capture the meaning of sentences in a well-defined format. One of the most prominent frameworks for achieving this is *Abstract Meaning Representation* (AMR) (Banarescu et al., 2013). In AMR, sentences are represented as directed acyclic and rooted graphs. An example is displayed in Figure 1, where we see three equivalent displays of an AMR that represents the meaning of the sentence “*The baby is sleeping well*”. In AMR, nodes are variables or concepts, while (labeled) edges express their relations. Among other phenomena, this allows AMR to capture coreference (via re-entrant structures) and semantic roles (via  $:arg_n$  relation). Furthermore, AMR links sentences to KBs: e.g., predicates are mapped to PropBank (Palmer et al., 2005; Kingsbury and Palmer, 2002), while named

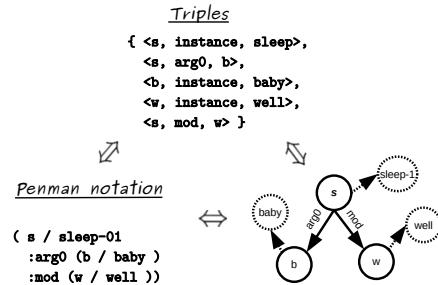


Figure 1: Equivalent representations of the AMR for “*The baby is sleeping well*”.

```
(p4 / possible-01
  :arg1 (d5 / destabilize-01
    :arg0 [:arg1] (c3 / country
      :quant (w2 / whole)))
  :condition (e1 / economy
    [:poss c3]
    :arg0-of (f0 / function-01
      [:pol -] )))
```

Figure 2: Parse of *Without a functioning economy, the whole country may destabilize* with errors outlined.

entities are linked to Wikipedia. From a logical perspective, AMR is closely related to first-order logic (FOL, see Bos (2016, 2019) for translation mechanisms).

Currently, AMRs are leveraged to enhance a variety of natural language understanding tasks. E.g., they have enhanced commonsense reasoning and question answering (Mitra and Baral, 2016), machine translation (Song et al., 2019), text summarization (Liao et al., 2018; Dohare et al., 2017) and paraphrasing (Issa et al., 2018). However, there is a critical issue with automatically generated AMRs (parses): they are often deficient.

These deficiencies can be quite severe, even when high-performance parsers are used. For example, in Figure 2, a neural parser (Lyu and Titov, 2018) conducts several errors when parsing *Without a functioning economy the whole country may destabilize*. E.g., it misses a negative polarity and classifies a patient argument as the agent by failing

graph representation	computer processing	human understanding	well-defined
triples	✓(e.g., GNN)	✗	✓
graph visualization	✗	✓(short sentences)	✗
PENMAN, linearized string	✓(e.g., LSTM)	✗	✓
PENMAN, indents	✓(this work)	✓	✓

Table 1: Four (equivalent) AMR representations and their accessibility with respect to human or computer (✓: ‘okay’, ✗: ‘perhaps possible, but difficult’).

to see that *destabilize* here functions as an ergative verb (parser: the country is the causer of destabilize; correct: the country is the object that is destabilized). In sum, the parse has misrepresented the sentence’s meaning.<sup>1</sup> However, assessing such deficiencies via comparison against a gold reference (as in classical parser evaluation) is often infeasible in practice: it takes a trained annotator and appr. 10 minutes to manually create one AMR (Banarescu et al., 2013).

To mitigate these issues, we would like to automatically rate the quality of AMRs without the costly gold graphs. This would allow us to signal downstream task systems the incorporated graphs’ trustworthiness or select among different candidate graphs from different parsing systems. To achieve this, we propose a method that imitates a human rater, who is inspecting the graphs. We show that the method can efficiently rate the quality of the AMRs in the absence of gold graphs.

The remainder of the paper is structured as follows: in Section 2, we outline our idea to exploit the textual multi-line string representation of AMRs, allowing for efficient and simple AMR processing while preserving vital graph structure. In Section 2.2, we instantiate this idea in a lightweight CNN that predicts the quality of AMR graphs along multiple dimensions of interest. In our experiments (Section 3), we show that this framework is efficient and performs better than strong baselines. Our code is available at <https://github.com/flipz357/amr-quality-rater>.

## 2 AMR as image with latent channels

In this section, we first motivate to treat AMRs as images with latent channels in order to rate them efficiently. Second, we briefly describe the task at hand: Rating the quality of AMR graphs in the absence of gold graphs. Finally, to solve this task, we create a lightweight CNN that evaluates AMR quality in multiple dimensions of interest.

<sup>1</sup>?With a functioning economy, the whole country may cause something to destabilize.

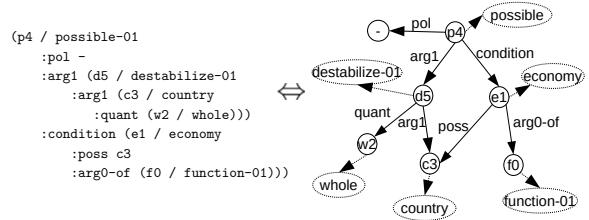


Figure 3: Different displays for an AMR structure of a sentence that has medium length (left: PENMAN notation, right: graphical visualization)

### The PENMAN notation and its (hidden) advantages

The native AMR notation is called PENMAN-notation or *Sentence Plan Language* (Kasper, 1989; Mann, 1983). Provably, an advantage of this notation is that it allows for secure AMR storage in text files. However, we argue that it has more advantages. For example, due to its clear structure, it allows humans a fairly quick understanding even of medium-sized to large AMR structures (Figure 3, left). On the other hand, we argue that a graphical visualization of such medium-sized to large AMRs (Figure 3, right) could hamper intuitive understanding, since the abundant visual signals (circles, arrows, etc.) may more easily overwhelm humans. Moreover, in every display, one would depend on an algorithm that needs to determine a suitable (and spacious) arrangement of the nodes, edges and edge labels. It may be for these reasons, that in the AMR annotation tool<sup>2</sup>, the graph that is under construction is always shown in PENMAN notation to the human user.

In sum, we find that the indented multi-line PENMAN form possesses three key advantages (Table 1): (i) it enables fairly easy human understanding, (ii), it is well-defined and (iii), which is what we will show next, it can be computationally exploited to better rate AMR quality.

### AMR as image to preserve graph structure

Figure 4 describes our proposed sentence representation treatment. After non-degenerate AMR graph simplification (more details in *Preprocessing*, 3.1), we first project the PENMAN representation onto a small grid (‘image’). Each AMR token (e.g., a node or an edge) is represented as a ‘categorical pixel’. Second,  $\Phi$  adds latent ‘channels’ to the categorical pixels, which can be learned incrementally in an application. In other words, every AMR token is represented by a fixed-sized vector of real

<sup>2</sup><https://www.isi.edu/cgi-bin/div3/mt/amr-editor/login-gen-v1.7.cgi>

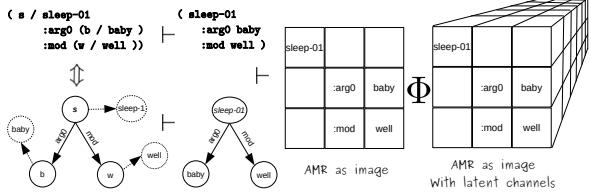


Figure 4: We transform the (simplified) PENMAN representation to an image and use  $\Phi$  to add latent channels.

numbers. These vectors are arranged such that the original graph structure is fully preserved.

## 2.1 Task: Rating the quality of AMR graphs

We aim at rating the quality of AMR graphs ('parses') in the absence of gold graphs. This boils down to answering the following question: *how well does a candidate AMR graph capture a given natural language sentence?* Therefore, the exact goal in this task is to learn a mapping

$$f : \mathcal{S} \times \mathcal{G} \rightarrow \mathbb{R}^d, \quad (1)$$

that maps a sentence  $s \in \mathcal{S}$  together with a candidate AMR graph  $g \in \mathcal{G}$  onto  $d$  scores, which describe the AMR with regard to  $d$  quality dimensions of interest. A successful mapping function should strongly correlate with the gold scores as they would emerge from evaluation against gold graphs. We proceed by describing the targeted dimensions in more detail.

**Main AMR quality dimensions** The main quality dimensions that we desire our model to predict are estimated **Smatch F1/recall/precision**. Smatch is the canonical AMR metric, assessing the triple overlap between two graphs, after an alignment step (Cai and Knight, 2013).

**AMR sub-task quality dimensions** However, we predict also other quality dimensions to assess various AMR aspects (Damonte et al., 2017). In this place, we can merely provide a brief overview: (i) **Unlabeled**: Smatch F1 when disregarding edge-labels. (ii) **No WSD**: Smatch F1 when ignoring ProbBank senses. (iii) **Frames**: PropBank frame identification F1 (iii) **Wikification**: KB linking F1 score on :wiki relations. (iv) **Negations**: negation detection F1. (v) **NamedEnt**: NER F1. (vi) **NS frames**: F1 score for ProbBank frame identification when disregarding the sense. (vii) **Concepts** F score for concept identification (viii) **SRL**: Smatch F1 computed on arg-i roles only. (ix) **Reentrancy**: Smatch F1 computed on re-entrant edges only. (x)

**IgnoreVars**: F1 when variable nodes are ignored.  
**(xi) Concepts**: F1 for concept detection.

## 2.2 A lightweight CNN to rate AMR quality

We want to model  $f$  (Eq. 1) in order to estimate a suite of quality scores  $y \in \mathbb{R}^d$  for any automatically generated AMR graph, given only the graph and the sentence from whence it is derived. Following Opitz and Frank (2019), we will contrast the AMR against the sentence's dependency parse, exploiting observed structural similarities between these two types of information (Wang et al., 2015). Our proposed method allows this in a simple way by processing dependency and AMR graphs in parallel. The architecture is outlined in Figure 5.

**Symbol embedding** The latent channels of AMR and dependency 'pixels' represent the embeddings of the 'tokens' or 'symbols' contained in the AMR and dependency vocabulary. These symbols represent nodes or edges. We use two special tokens: the <tab> token, which represents the indentation level, and the <pad> token, which fills the remaining empty 'pixels'. By embedding lookup, we obtain AMR and dependency images with 128 latent channels and 45x15 'pixels' ( $\Phi$  in Figure 5; the amount of pixels is chosen such that more than 95% of training AMRs can be fully captured).

**Encoding local graph regions** Given AMR and dependency images with 128 latent channels and 45x15 pixels, we apply to each of the two images 256 filters of size 3x3, which is a standard type of kernel in CNNs. This converts both graphs to 256 feature maps each  $\in \mathbb{R}^{45 \times 15}$  (same-padding), obtaining two three-dimensional tensors  $L_{amr}^1, L_{dep}^1 \in \mathbb{R}^{45 \times 15 \times 256}$ . From here, we construct our first joint representation, which matches local dependency regions with local AMR regions:

$$j_{res} = GPF(L_{amr}^1 \otimes L_{dep}^1), \quad (2)$$

where  $x \otimes y = [x \odot y; x \ominus y]$  denotes the concatenation of element-wise multiplication and element-wise subtraction.  $GPF$  is an operation that performs global pooling and vectorization ('flattening') of any input tensor. This means that  $j_{res} \in \mathbb{R}^{512}$  is a joint representation of the locally matched dependency and AMR graph regions. This intermediate process is outlined in Figure 5 by  $\otimes$  (left) and  $GPF$ . Finally, we reduce the dimensions of the two intermediate three-dimensional representations  $L_{amr}^1$  and  $L_{dep}^1$  with 3x3 max-pooling and

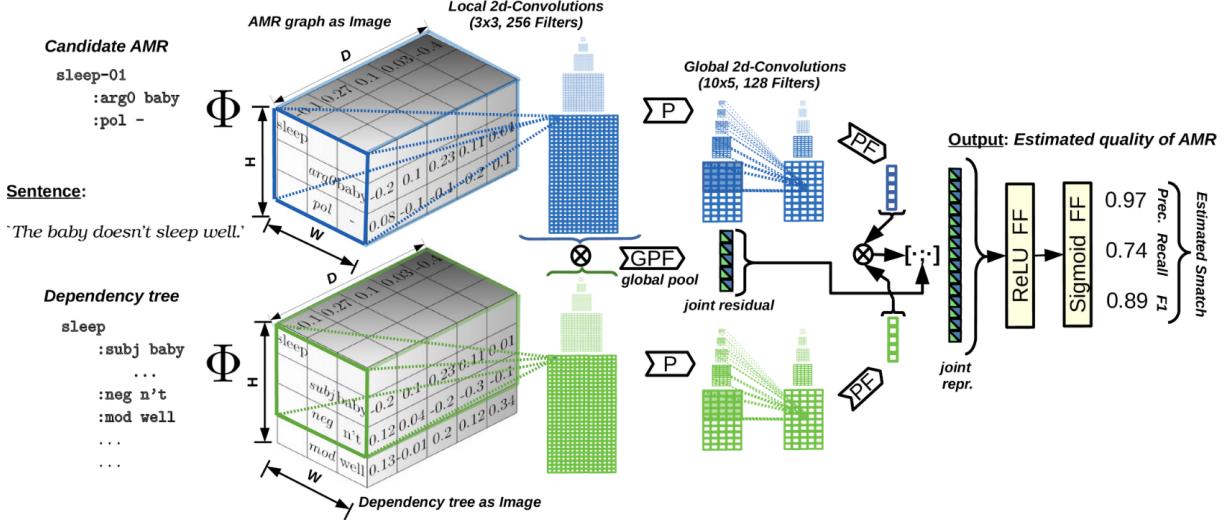


Figure 5: Our proposed architecture for efficient AMR quality assessment.

obtain  $L_{amr}^2$  and  $L_{dep}^2 \in \mathbb{R}^{15 \times 5 \times 256}$

**Encoding global graph regions** For a moment, we put the *joint residual* ( $j_{res}$ ) aside and proceed by processing the locally convolved feature maps with larger filters. While the first convolutions allowed us to obtain abstract *local* graph regions  $L_{amr}^2$  and  $L_{dep}^2$ , we now aim at matching more *global* regions. More precisely, we use 128 2D filters of shape 10x5, followed by a 5x5 max-pooling operations on  $L_{amr}^2$  and  $L_{dep}^2$ . Thus, we have obtained vectorized abstract global graph representations  $g_{amr}, g_{dep} \in \mathbb{R}^{384}$ . Then, we construct a joint representation (right  $\otimes$ , Figure 5):

$$j_{glob} = g_{amr} \otimes g_{dep}. \quad (3)$$

At this point, together with the joint residual representation from the local region matching, we have arrived at two joint vector representations  $j_{glob}$  and  $j_{res}$ . We concatenate them ( $[; \cdot]$  in Figure 5) to form one joint representation  $j \in \mathbb{R}^{1280}$ :

$$j = [j_{res}; j_{glob}] \quad (4)$$

**Quality prediction** The shared representation  $j$  is further processed by a feed-forward layer with ReLU activation functions ( $FF_{+ReLU}$ , Figure 5) and a consecutive feed-forward layer with sigmoid activation functions ( $FF_{+sigm}$ , Figure 5):

$$y = \text{sigm}(ReLU(j^T A)B), \quad (5)$$

where  $A \in \mathbb{R}^{1280 \times h}$ ,  $B \in \mathbb{R}^{h \times \dim(out)}$  are parameters of the model and  $\text{sigm}(x) = (\frac{1}{1+e^{-x_1}}, \dots, \frac{1}{1+e^{-x_{\dim(out)}}})$  projects  $x$  onto

$[0, 1]^{\dim(out)}$ . When estimating the main AMR metric scores we instantiate three output neurons ( $\dim(out) = 3$ ) that represent estimated Smatch precision, Smatch recall and Smatch F1. In the case where we are interested in a more fine-grained assessment of AMR quality (e.g., knowledge-base linking quality), we have 33 output neurons representing expected scores for various semantic aspects involved in AMR parsing (we predict precision, recall and F1 of 11 aspects, as outlined in §2.1).

To summarize, the residual joint representation should capture local similarities. On the other hand, the second joint representation aims to capture the more global and structural properties of the two graphs. Both types of information inform the final quality assessment of our model in the last layer.

### 3 Experiments

In this section, we first describe the data, changes to the data that target the reduction of biases, and the baseline. After discussing our main results, we conduct further analyses. (i), we study the effects of our data-debiasing steps. (ii), we assess the performance of our model in a classification task (distinguishing good from bad parses). (iii), we assess the model performance when we only provide the candidate AMR and the sentence (dependency tree ablation). (iv), we provide detailed measurements of the method’s computational cost.

#### 3.1 Experimental setup

**Data** We use the data from Opitz and Frank (2019). The data set consists of more than 15,000

sentences with more than 60,000 corresponding parses, by three different automatic parsing systems and a human. More precisely, the data set  $\mathcal{D} = \{(s_i, g_i, y_i)\}_{i=1}^N$  consists of tuples  $(s_i, g_i, y_i)$ , where  $s_i \in \mathcal{S}$  is a natural language sentence,  $g_i \in \mathcal{G}$  is a ‘candidate’ AMR graph and  $y_i \in \mathbb{R}^d$  is a 36-dimensional vector containing scores which represent the quality of the AMR graph in terms of precision, recall and F1 with respect to 12 different tasks captured by AMR (as outlined in §2.1).

**Debiasing of the data** We observe three biases in the data. First, the graphs in the training section of our data are less deficient than in the development and testing data, because the parsers were trained on (*sentence, gold graph*) pairs from the training section. For our task, this means that the training section’s target scores are higher, on average, than the target scores in the other data partitions. To achieve more balance in this regard, we re-split the data randomly on the sentence-id level (such that a sentence does not appear in more than one partition with different parses).

Second, we observe that the data contains some superficial hidden clues that could give away the parse’s source. This bears the danger that a model does not learn to assess the *parse quality*, but to assess the *source of the parse*. And since some parsers are better or worse than others, the model could exploit this bias. For example, consider that one parser prefers to write  $(r / run-01 :arg1 (c / cat) :polarity - )$ , while the other parser prefers to write  $(r / run-01 :polarity - :arg1 (c / cat))$ . These two structures are semantically equivalent but differ on the surface. Hence, the arrangement of the output may provide unwanted clues on the source of the parse. To alleviate this issue, we randomly re-arrange all parses on the surface, keeping their semantics.<sup>34</sup>

A third bias stems from a design choice in the metric scripts used to calculate the target scores. More precisely, the extended *Smatch-metric* script, per default, assigns a parse that does not contain a certain edge-type (e.g.,  $:arg_n$ ) the score 0 with respect to the specific quality dimension (in this case, SRL: 0.00 Precision/Recall/F1). However, if the gold parse also does not contain an

<sup>3</sup>Technically, this is achieved by reformatting the parses such that in the depth-first writing-traversal at node  $n$  the out-going edges of  $n$  will be traversed in random order.

<sup>4</sup>Different variable names, e.g.,  $(r / run-01)$  and  $(x / run-01)$  are not an issue in this work since the variables are handled via (van Noord and Bos, 2017a). See also *Preprocessing*, §3.1

edge of this type (i.e.,  $:arg_n$ ), then we believe that the correct default score should be 1, since the parse is, in the specific dimension, in perfect agreement with the gold (i.e., SRL: 1.00 Precision/Recall/F1). Therefore, we set all sub-task scores, where the predicted graph agrees with the gold graph in the absence of a feature, from 0 to 1.

**Preprocessing** Same as prior work, we dependency-parse and tokenize the sentences with spacy (Honnibal and Montani, 2017) and replace variables with corresponding concepts (e.g.,  $(j / jump-01 :arg0 (g / girl))$  is translated to  $(jump-01 :arg0 (girl))$ ). Re-entrancies are handled with pointers according to van Noord and Bos (2017a), which ensures non-degenerate AMR simplification.<sup>5</sup> Furthermore, we lower-case all tokens, remove quotation marks and join sub-structures that represent names.<sup>6</sup> The vocabulary encompasses all tokens of frequency  $\geq 5$ , remaining ones are set to  $<\text{unk}>$ .

**Training** All parameters are initialized randomly. We train for 5 epochs and select the parameters  $\theta$  from the epoch where maximum development scores were achieved (with respect to average Pearson’s  $\rho$  over the quality dimensions). In training, we reduce the squared error with gradient descent (Adam rule (Kingma and Ba, 2019), learning rate = 0.001, mini batch size = 64):

$$\theta^* = \arg \min_{\theta} \sum_{i=1}^{|\mathcal{D}|} \sum_{j=1}^{|M|} (y_{i,j} - f_{\theta}(s_i, g_i)_j)^2, \quad (6)$$

where  $M$  is the set of target metrics.

**Baseline** Our main baseline is the model of previous work, henceforth denoted by **LG-LSTM**. The method works in the following steps: first, it uses a depth-first graph traversal to linearize the automatic AMR graph and the corresponding dependency tree of the sentence. Second, it constructs a joint representation and predicts the score estimations. To further improve its performance, the baseline uses some extra-features (e.g., a shallow alignment from

<sup>5</sup>For example, consider the sentence *The cat scratches itself* and its graph  $(x / scratch-01 :arg0 (y / cat) :arg1 y)$ . Replacing the variables with concepts would come at the cost of an information loss w.r.t. to coreference:  $(scratch-01 :arg0 cat :arg1 cat)$  — does the cat scratch itself or another cat? Hence, pointers are used to translate the graph into  $(scratch-01 :arg0 *0* cat :arg1 *0*)$ .

<sup>6</sup>E.g.,  $:name (name :op1 ‘Barack’ :op2 ‘Obama’)$  is translated to  $:name barack obama$ .

	Smatch	Ridge	GNN	LG-LSTM	ours	change %
P's $\rho$	F1	0.428	0.659	0.662 $\pm$ 0.00	0.696 $\pm$ 0.00	+5.14 ‡‡
	Precision	0.348	0.601	0.600 $\pm$ 0.00	0.623 $\pm$ 0.01	+3.83 †
	Recall	0.463	0.667	0.676 $\pm$ 0.00	0.719 $\pm$ 0.00	+6.36 ‡‡
RMSE	F1	0.155	0.132	0.130 $\pm$ 0.00	0.128 $\pm$ 0.00	-1.54
	Precision	0.146	0.127	0.126 $\pm$ 0.00	0.126 $\pm$ 0.00	+0.0
	Recall	0.169	0.141	0.142 $\pm$ 0.00	0.136 $\pm$ 0.00	-4.23

Table 2: Main results. Pearson’s corr. coefficient (row 1-3) is better if higher; root mean square error (RMSE, row 4-6) is better if lower. The quality dimensions are explained in §2.1. † (‡):  $p < 0.05$  ( $p < 0.005$ ), significant difference in the correlations with two-tailed test using Fisher  $\rho$  to z transformation (Fisher, 1915).

dependency tokens to AMR tokens).<sup>7</sup> Generally speaking, the baseline is a model that works based on graph linearizations. Such type of model, despite its apparent simplicity, has proven to be an effective baseline or state-of-the-art method in various works about converting texts into graphs (Konstas et al., 2017; van Noord and Bos, 2017b), or converting graphs into texts (Bastings et al., 2017; Beck et al., 2018; Song, 2019; Pourdamghani et al., 2016; Song et al., 2018; Vinyals et al., 2015; Mager et al., 2020), or performing mathematically complex tasks modeled as graph-to-graph problems, such as symbolic integration (Lample and Charton, 2020). However, in our main results, we also display the results of two additional baselines: **GNN** (Song et al., 2018), where we encode the dependency tree and the AMR with a graph-recurrent encoder and perform regression on the joint averaged node embedding vectors.<sup>8</sup> And **Ridge**, an l2-regularized linear regression that is based on shallow graph statistics.<sup>9</sup>

## 3.2 Results

**Main AMR quality dimensions** The main quality of an AMR graph is estimated in expected triple match ratios (Smatch F1, Precision and Recall). The results, averaged over 10 runs, are displayed in Table 2. With regard to estimated Smatch F1, we

<sup>7</sup>Furthermore, the baseline uses auxiliary losses to achieve a slight performance gain in predicting the Smatch metrics. For the sake of simplicity, we do not use these auxiliary losses, except in one experiment, where we show that our method achieves a similar small gain with the auxiliary losses.

<sup>8</sup> $\left[ \frac{1}{|V_A|} \sum_{v \in V_A} emb(v) \right] \otimes \left[ \frac{1}{|V_D|} \sum_{v \in V_D} emb(v) \right]$ .

<sup>9</sup>For the dependency graph (D) and the AMR graph (A) we both compute  $\phi(A|D) = [\text{density}, \text{avg. node degree}, \text{node count}, \text{edge count}, (\text{arg0}|\text{subj}) \text{ count}, (\text{arg1}|\text{obj}) \text{ count}]$ , the final feature vector then is defined as  $\Phi(x) = [\phi(A) - \phi(D); \phi(D); \phi(A); \frac{|\text{lemmas}(D) \cap \text{concepts}(A)|}{|\text{lemmas}(D) \cup \text{concepts}(A)|}]$

	Quality Dim.	LG-LSTM	ours	change %
Pearson's $\rho$	Concepts	0.508 $\pm$ 0.01	0.545 $\pm$ 0.01	+7.28 †
	Frames	0.420 $\pm$ 0.01	0.488 $\pm$ 0.01	+16.19 ‡‡
	IgnoreVars	0.627 $\pm$ 0.01	0.665 $\pm$ 0.00	+6.06 ‡‡
F1 Pearson's $\rho$	NamedEnt.	0.429 $\pm$ 0.02	0.460 $\pm$ 0.01	+7.23 †
	Negations	0.685 $\pm$ 0.02	0.746 $\pm$ 0.01	+8.91 ‡‡
	NoWSD	0.640 $\pm$ 0.01	0.680 $\pm$ 0.00	+6.25 ‡‡
F1 RMSE	NS-frames	0.419 $\pm$ 0.02	0.505 $\pm$ 0.01	+20.53 ‡‡
	Reentrancies	0.508 $\pm$ 0.01	0.602 $\pm$ 0.00	+18.50 ‡‡
	SRL	0.519 $\pm$ 0.01	0.581 $\pm$ 0.01	+11.95 ‡‡
Unlabeled	Wikification	0.901 $\pm$ 0.00	0.904 $\pm$ 0.00	+0.33
	Concepts	0.117 $\pm$ 0.00	0.114 $\pm$ 0.00	-2.56
	Frames	0.186 $\pm$ 0.00	0.182 $\pm$ 0.00	-2.15
F1 RMSE	IgnoreVars	0.195 $\pm$ 0.00	0.186 $\pm$ 0.00	-4.62
	NamedEnt.	0.159 $\pm$ 0.00	0.156 $\pm$ 0.00	-1.89
	Negations	0.197 $\pm$ 0.00	0.180 $\pm$ 0.00	-8.63
F1 RMSE	NoWSD	0.132 $\pm$ 0.00	0.126 $\pm$ 0.00	-4.55
	NS-frames	0.157 $\pm$ 0.00	0.155 $\pm$ 0.00	-1.27
	Reentrancies	0.285 $\pm$ 0.00	0.265 $\pm$ 0.00	-7.02
SRL	SRL	0.189 $\pm$ 0.00	0.181 $\pm$ 0.00	-4.23
	Unlabeled	0.124 $\pm$ 0.00	0.121 $\pm$ 0.00	-2.42
	Wikification	0.165 $\pm$ 0.00	0.162 $\pm$ 0.00	-1.82

Table 3: Results for AMR quality rating w.r.t. various sub-tasks. † (‡): significance (c.f. caption Table 2).

achieve a correlation with the gold scores of 0.695 Pearson’s  $\rho$ . This constitutes a significant improvement of appr. 5% over LG-LSTM. Similarly, recall and precision correlations improve by 6.36% and 3.83 % (from 0.676 to 0.719 and 0.600 to 0.623). While the improvement in predicted recall is significant at  $p < 0.05$  and  $p < 0.005$ , the improvement in predicted precision is significant at  $p < 0.05$ . When we consider the root mean square error (RMSE), we find that the method improves over the best baseline by -1.54% in estimated Smatch F1 and -4.23% in estimated Smatch recall. On the other hand, the RMS error in estimated precision remains unchanged.

**AMR subtask quality** Our model can also rate the quality of an AMR graph in a more fine-grained way. The results are displayed in Table 3. Over almost every dimension we see considerable improvements. For instance, a considerable improvement in Pearson’s  $\rho$  is achieved for assessment of *frame prediction quality* (‘NSFrames’ in Table 3, +20.5%  $\rho$ ) and *coreference quality* (‘Reentrancies’ in Table 3, +18.5%).

A substantial error reduction is achieved in *polarity* (‘Negations’, Table 3), where we reduce the RMSE of the estimated F1 score by -8.6%. When rating the SRL-quality of an AMR parse, our model reduces the RMSE by appr. 4%. In general,

data	method	Pearson’s $\rho$			error RMSE (F1)
		P	R	F1	
$\frac{0}{2}$	LG-LSTM	0.72	0.78	0.77	0.138
	LG-LSTM <sub>+aux</sub>	0.74	0.79	0.78	0.137
	ours	0.75	0.80	0.79	0.133
	ours <sub>+aux</sub>	<b>0.76</b>	<b>0.81</b>	<b>0.80</b>	<b>0.132</b>
$\frac{1}{2}$	LG-LSTM	0.67	0.73	0.72	0.120
	ours	<b>0.68</b>	<b>0.75</b>	<b>0.74</b>	<b>0.117</b>
$\frac{2}{2}$	LG-LSTM	0.60	0.68	0.66	0.130
	ours	<b>0.62</b>	<b>0.72</b>	<b>0.70</b>	<b>0.128</b>

Table 4: Performance-effects of data debiasing steps.  $+_{aux}$  indicates a model variant that is trained using auxiliary losses that incorporate information about the other AMR aspects in the training process (see Fn.<sup>7</sup>).

improvements are obtained over almost all tested quality dimensions, both in RMSE reduction and increased correlation with the gold scores.

### 3.3 Analysis

**Effect of data debiasing** We want to study the effect of the data set cleaning steps by analyzing the performance of our method and the baseline on three different versions of the data, with respect to estimated Smatch scores. The three versions are (i)  $\frac{0}{2} = \text{AMRQUALITY}$ , which is the original data; (ii)  $\frac{1}{2}$ , which is the data after the random re-split and score correction; (iii)  $\frac{2}{2} = \text{AMRQUALITYCLEAN}$  which is our main data after the final debiasing step (shallow structure debiasing) has been applied.

The results are shown in Table 4. We can make three main observations: (i) from the first to the second debiasing step, the baseline and our model have in common that Pearson’s  $\rho$  and the error decrease. While we cannot exactly explain why  $\rho$  decreases, it is somewhat in line with recent research that observed performance drops when data was re-split (Gorman and Bedrick, 2019). On the other hand, the error decrease can be explained by the random re-split that balances the target scores. (ii) The second debiasing step leads to a decrease in  $\rho$  and an increase in error, for both models. This indicates that we have successfully removed shallow biases from the data that can give away the parse’s source. (iii) On all considered versions of the data, the method performs better than the baseline.

**AMRs: telling the good from the bad** In this experiment, we want to see how well the model can discriminate between good and bad graphs. To this aim, we create a five-way classification task: graphs are assigned the label ‘very bad’ (Smatch F1  $< 0.25$ ), ‘bad’ ( $0.25 \geq \text{Smatch F1} < 0.5$ ), ‘good’

	majority	random	LG-LSTM	ours
avg. F1	0.13	0.20	0.40	0.44 <sup>†‡</sup>
quadr. kappa	0.0	0.03	0.53	0.60 <sup>†‡</sup>

Table 5: Graph quality classification task.  $\dagger$  ( $\ddagger$ ) significance with paired t-test at  $p < 0.05$  ( $p < 0.005$ ) over 10 random initializations.

	Quality Dim.	LG-LSTM	ours	ours (no dep.)
$P_s \rho$	Smatch F1	$0.662 \pm 0.00$	$0.696 \pm 0.00$	$0.682 \pm 0.01$
	Smatch precision	$0.600 \pm 0.00$	$0.623 \pm 0.01$	$0.614 \pm 0.01$
	Smatch recall	$0.676 \pm 0.00$	$0.719 \pm 0.00$	$0.702 \pm 0.01$
RMSE	Smatch F1	$0.130 \pm 0.00$	$0.128 \pm 0.00$	$0.128 \pm 0.00$
	Smatch precision	$0.126 \pm 0.00$	$0.126 \pm 0.00$	$0.129 \pm 0.00$
	Smatch recall	$0.142 \pm 0.00$	$0.136 \pm 0.00$	$0.139 \pm 0.00$

Table 6: Right column: results of our system when we abstain from feeding the dependency tree, and only show the sentence together with the candidate AMR.

( $0.5 \geq \text{Smatch F1} < 0.75$ ), ‘very good’ ( $0.75 \geq \text{Smatch F1} < 0.95$ ) and ‘excellent’ ( $\text{Smatch F1} \geq 0.95$ ). Here, we do not retrain the models with a classification objective but convert the estimated Smatch F1 to the corresponding label. Since the classes are situated on a nominal scale, and ordinary classification metrics would not fully reflect the performance, we also use quadratic weighted kappa (Cohen, 1968) for evaluation.

The results are shown in Table 5. All baselines, including LG-LSTM, are significantly outperformed by our approach, both in terms of macro F1 (+4 points, 10% improvement) and quadratic kappa (+7 points, 13% improvement).

**How important is the dependency information?** To investigate this question, instead of feeding the dependency tree of the sentence, we only feed the sentence itself. To achieve this, we simply insert the tokens in the first row of the former dependency input image, and pad all remaining empty ‘pixels’. In this mode, the sentence encoding is similar to standard convolutional sentence encoders as they are typically used in many tasks (Kim, 2014).

The results are shown in the right column of Table 6. The performance drops are small but consistent across all analyzed dimensions, both in terms of error (0 to 2.2% increase) and Pearson’s  $\rho$  (1.4 to 2.4% decrease). This indicates that the dependency trees contain information that can be exploited by our model to better judge the AMR quality. We hypothesize that this is due to similarities between relations such as *subj/obj* (syntactic) or *arg0/arg1* (semantic), etc. Yet, we see that this simpler model,

GPU type	GTX Titan		GTX 1080	
method	LG-LSTM	ours	LG-LSTM	ours
avg. ep. time	722s	59s	1582s	64s
avg. W	105	166	45	128
kWh per epoch	0.021	0.003	0.020	0.002

Table 7: Efficiency analysis of two approaches.

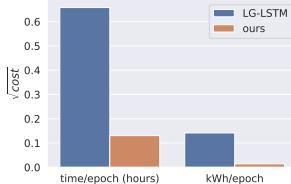


Figure 6: Training cost diagram of two approaches.

which does not see the dependency tree, still outperforms the baseline, except in estimated precision, where the error is increased by 2.4%.

**Efficiency analysis** Recently, in many countries, there have been efforts to reduce energy consumption and carbon emission. Since deep learning typically requires intensive GPU computing, this aspect is of increasing importance to researchers and applicants (Strubell et al., 2019; Tang et al., 2019; Ganguly et al., 2019). To investigate energy consumption of our method and previous work, we monitor their GPU usage during training, assessing the following quantities : (i) avg. time per epoch, (ii) avg. watts GPU usage, (iii) kilowatts per epoch (in kWh).

The results of this analysis are displayed in Table 7 and outlined in Figure 6. Our method consumes approximately 6.6 times less total kWh on a GTX Titan (10 times less on a GTX 1080). Directly related, it also reduces the training time: prior work requires appr. 1500s training time per epoch (GTX 1080), while our method requires appr. 60s per epoch (GTX 1080). The main reason for this is that our model does not depend on recurrent operations and profits more from parallelism.

## 4 Related work

**Quality measurement of structured predictions** Since evaluating structured representations against human annotations is costly, systems have been developed that attempt an automatic quality assessment of these structures. Due to its popularity, much work has been conducted in *machine translation* (MT) under the umbrella of *quality estimation* (QE). QE can take place either on a word-level (Martins et al., 2017), sentence-level (Spe-

cia et al., 2009), or document-level (Scarton et al., 2015). The conference on Machine Translation (WMT) has a long-standing workshop and shared task-series on MT quality assessment (Bojar et al., 2013, 2014, 2015, 2016, 2017; Specia et al., 2018; Fonseca et al., 2019). Quality estimation for neural language generation has been investigated, i.a., by Scarton et al. (2016), and recently by Dušek et al. (2019), who design a model that jointly learns to rate and rank generations, or by Zopf (2018), who predicts pair-wise preferences for generated summaries.

Furthermore, automatic techniques for the quality assessment of syntactic parses have been proposed. For instance, Ravi et al. (2008) formulate the task as a single-variable regression problem to assess the quality of constituency trees. A major difference to our work is that they try to assess the performance of a *single parser*, while we aim at a parser-agnostic setting where candidate parses stem from *different parsers*. Similarly, Kawahara and Uchimoto (2008) predict a binary label that reflects whether the tree-quality lies above a certain threshold (or not). When multiple candidate parses are available, tree ranking methods (Zhu et al., 2015; Zhou et al., 2016) may also be interpreted as some form of parse quality assessment (see Do and Rehbein (2020) for a recent overview). Compared with assessing the quality of (abstract) meaning representations, judging about syntactic trees perhaps is a conceptually slightly simpler task, since the syntactic graphs are more directly grounded in the sentence<sup>10</sup>, and therefore it may be easier to judge whether graph components are correct, redundant, missing, or false.

In comparison to MT, automatic quality assessment of meaning representations is insufficiently researched. Opitz and Frank (2019) propose an LSTM based model that performs a multi-variate quality analysis of AMRs (constituting the baseline which we compared against). We believe that quality estimation approaches may also prove valuable for other meaning representation formalisms (MRs), such as, e.g., discourse representations (Kamp and Reyle, 1993; Kamp, 2008; Abzianidze et al., 2019) or universal semantic dependencies (Reisinger et al., 2015; Stengel-Eskin et al., 2020). For example, since the manual creation of MRs

<sup>10</sup>In dependency trees, nodes are words; in constituency trees, nodes are (labeled) phrases; in meaning representations, words or phrases may be projected to abstract semantic nodes, or they may be omitted.

is a notoriously laborious task, automatic quality assessment tools could assist humans in the annotation process (e.g., by serving as a cheap annotation quality check or by filtering automatic parses in active learning).

**AMR metrics** When a gold graph is available, it can be used to compute the canonical AMR metric Smatch (Cai and Knight, 2013) that assesses matching triples. Furthermore, Damonte et al. (2017) have extended Smatch to inspect various aspects of AMR. In this work, we have shown that our model can predict the expected outcomes of these metrics in the absence of the gold graph. Recently, more AMR metrics have been proposed, for example the Bleu-based (Papineni et al., 2002) SemBleu metric (Song and Gildea, 2019), Sema (Anchieta et al., 2019) or S<sup>2</sup>match (Opitz et al., 2020), a variant of Smatch. We plan to extend our model such that it also predicts these metrics.

**AMR parsing** Recent advances in AMR parsing have been achieved by parsers that either predict latent alignments jointly with nodes (Lyu and Titov, 2018), or by transducing a graph from a sequence with a minimum spanning tree (MST) decoding algorithm (Zhang et al., 2019), or by focusing on core semantics in a top-down fashion (Cai and Lam, 2019), or by performing auto-regressive decoding with a graph encoder (Cai and Lam, 2020). Other approaches apply statistical machine translation (Pust et al., 2015) or sequence-to-sequence models, which tend to suffer from data scarcity issues and need considerable amounts of silver data to improve results (van Noord and Bos, 2017c; Konstas et al., 2017). Previously, alignment-based pipeline models have proved effective (Flanigan et al., 2014) or transition-based approaches that convert dependency trees step-by-step to AMR graphs (Wang et al., 2015, 2016; Lindemann et al., 2020).

## 5 Conclusion

In this work, we have developed an approach to rate the quality of AMR graphs in the absence of costly gold data. Our model imitates a human judge that is confronted, ‘on paper’, with the AMR in its native multi-line Penman format. We saw how this setup allowed efficient AMR processing with convolutions. Our experiments indicate that the method rates AMR quality more accurately and more efficiently than previous work.

## Acknowledgments

I am grateful to the anonymous reviewers for their valuable thoughts and comments. Moreover, I am grateful to Anette Frank for her thoughtful feedback on an earlier draft of this paper and her general guidance throughout my studies.

## References

- Lasha Abzianidze, Rik van Noord, Hessel Haagsma, and Johan Bos. 2019. [The first shared task on discourse representation structure parsing](#). In *Proceedings of the IWCS Shared Task on Semantic Parsing*, Gothenburg, Sweden. Association for Computational Linguistics.
- Rafael Torres Anchieta, Marco Antonio Sobrevilla Cabezudo, and Thiago Alexandre Salgueiro Pardo. 2019. [Sema: an extended semantic evaluation for amr](#). In *(To appear) Proceedings of the 20th Computational Linguistics and Intelligent Text Processing*. Springer International Publishing.
- Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. [Abstract meaning representation for sembanking](#). In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186.
- Joost Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Sima'an. 2017. [Graph convolutional encoders for syntax-aware neural machine translation](#). In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1957–1967, Copenhagen, Denmark. Association for Computational Linguistics.
- Daniel Beck, Gholamreza Haffari, and Trevor Cohn. 2018. [Graph-to-sequence learning using gated graph neural networks](#). In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 273–283, Melbourne, Australia. Association for Computational Linguistics.
- Ondřej Bojar, Christian Buck, Chris Callison-Burch, Christian Federmann, Barry Haddow, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2013. [Findings of the 2013 Workshop on Statistical Machine Translation](#). In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 1–44, Sofia, Bulgaria. Association for Computational Linguistics.
- Ondřej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Hervé Saint-Amand, Radu Soricut, Lucia Specia, and Aleš Tamchyna. 2014. [Findings of the 2014 workshop on statistical machine translation](#). In *Proceedings of the*

- Ninth Workshop on Statistical Machine Translation, pages 12–58, Baltimore, Maryland, USA. Association for Computational Linguistics.
- Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Shujian Huang, Matthias Huck, Philipp Koehn, Qun Liu, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Raphael Rubino, Lucia Specia, and Marco Turchi. 2017. *Findings of the 2017 conference on machine translation (WMT17)*. In *Proceedings of the Second Conference on Machine Translation*, pages 169–214, Copenhagen, Denmark. Association for Computational Linguistics.
- Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Aurélie Névéol, Mariana Neves, Martin Popel, Matt Post, Raphael Rubino, Carolina Scarton, Lucia Specia, Marco Turchi, Karin Verspoor, and Marcos Zampieri. 2016. *Findings of the 2016 conference on machine translation*. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 131–198, Berlin, Germany. Association for Computational Linguistics.
- Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Barry Haddow, Matthias Huck, Chris Hokamp, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Carolina Scarton, Lucia Specia, and Marco Turchi. 2015. *Findings of the 2015 workshop on statistical machine translation*. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 1–46, Lisbon, Portugal. Association for Computational Linguistics.
- Johan Bos. 2016. Expressive power of abstract meaning representations. *Computational Linguistics*, 42(3):527–535.
- Johan Bos. 2019. Separating argument structure from logical structure in amr. *arXiv preprint arXiv:1908.01355*.
- Deng Cai and Wai Lam. 2019. Core semantic first: A top-down approach for AMR parsing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3797–3807, Hong Kong, China. Association for Computational Linguistics.
- Deng Cai and Wai Lam. 2020. Amr parsing via graph-sequence iterative inference. *arXiv preprint arXiv:2004.05572*.
- Shu Cai and Kevin Knight. 2013. Smatch: an evaluation metric for semantic feature structures. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 748–752, Sofia, Bulgaria. Association for Computational Linguistics.
- Jacob Cohen. 1968. Weighted kappa: nominal scale agreement with provision for scaled disagreement or partial credit. *Psychological bulletin*, pages 213–220.
- Marco Damonte, Shay B. Cohen, and Giorgio Satta. 2017. An incremental parser for abstract meaning representation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 536–546, Valencia, Spain. Association for Computational Linguistics.
- Bich-Ngoc Do and Ines Rehbein. 2020. Neural reranking for dependency parsing: An evaluation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4123–4133, Online. Association for Computational Linguistics.
- Shibhang Dohare, Harish Karnick, and Vivek Gupta. 2017. Text summarization using abstract meaning representation. *arXiv preprint arXiv:1706.01678*.
- Ondřej Dušek, Karin Sevgnani, Ioannis Konstas, and Verena Rieser. 2019. Automatic quality estimation for natural language generation: Ranting (jointly rating and ranking). In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 369–376, Tokyo, Japan. Association for Computational Linguistics.
- Ronald A Fisher. 1915. Frequency distribution of the values of the correlation coefficient in samples from an indefinitely large population. *Biometrika*, 10(4):507–521.
- Jeffrey Flanigan, Sam Thomson, Jaime Carbonell, Chris Dyer, and Noah A. Smith. 2014. A discriminative graph-based parser for the abstract meaning representation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1426–1436, Baltimore, Maryland. Association for Computational Linguistics.
- Erick Fonseca, Lisa Yankovskaya, André F. T. Martins, Mark Fishel, and Christian Federmann. 2019. Findings of the WMT 2019 shared tasks on quality estimation. In *Proceedings of the Fourth Conference on Machine Translation (Volume 3: Shared Task Papers, Day 2)*, pages 1–10, Florence, Italy. Association for Computational Linguistics.
- A. Ganguly, R. Muralidhar, and V. Singh. 2019. Towards energy efficient non-von neumann architectures for deep learning. In *20th International Symposium on Quality Electronic Design (ISQED)*, pages 335–342.
- Kyle Gorman and Steven Bedrick. 2019. We need to talk about standard splits. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2786–2791, Florence, Italy. Association for Computational Linguistics.

- Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.
- Fuad Issa, Marco Damonte, Shay B. Cohen, Xiaohui Yan, and Yi Chang. 2018. *Abstract meaning representation for paraphrase detection*. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 442–452, New Orleans, Louisiana. Association for Computational Linguistics.
- Hans Kamp. 2008. *A Theory of Truth and Semantic Representation*, chapter 8. John Wiley & Sons, Ltd.
- Hans Kamp and Uwe Reyle. 1993. *From Discourse to Logic. Introduction to Model-theoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*. Kluwer, Dordrecht.
- Robert T. Kasper. 1989. A flexible interface for linking applications to penman’s sentence generator. In *Proceedings of the Workshop on Speech and Natural Language*, HLT ’89, pages 153–158, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Daisuke Kawahara and Kiyotaka Uchimoto. 2008. Learning reliability of parses for domain adaptation of dependency parsing. In *Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-II*.
- Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.
- Diederik P Kingma and J Adam Ba. 2019. A method for stochastic optimization. arxiv 2014. *arXiv preprint arXiv:1412.6980*, 434.
- Paul Kingsbury and Martha Palmer. 2002. *From TreeBank to PropBank*. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC’02)*, Las Palmas, Canary Islands - Spain. European Language Resources Association (ELRA).
- Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. 2017. *Neural AMR: Sequence-to-sequence models for parsing and generation*. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 146–157, Vancouver, Canada. Association for Computational Linguistics.
- Guillaume Lample and François Charton. 2020. Deep learning for symbolic mathematics. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.
- Kexin Liao, Logan Lebanoff, and Fei Liu. 2018. Abstract meaning representation for multi-document summarization. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1178–1190, Santa Fe, New Mexico, USA. Association for Computational Linguistics.
- Matthias Lindemann, Jonas Grotzschitz, and Alexander Koller. 2020. Fast semantic parsing with well-typedness guarantees. *arXiv preprint arXiv:2009.07365*.
- Chunchuan Lyu and Ivan Titov. 2018. AMR parsing as graph prediction with latent alignment. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 397–407, Melbourne, Australia. Association for Computational Linguistics.
- Manuel Mager, Ramón Fernandez Astudillo, Tahira Naseem, Md Arifat Sultan, Young-Suk Lee, Radu Florian, and Salim Roukos. 2020. GPT-too: A language-model-first approach for AMR-to-text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1846–1852, Online. Association for Computational Linguistics.
- William C. Mann. 1983. An overview of the penman text generation system. In *Proceedings of the Third AAAI Conference on Artificial Intelligence, AAAI’83*, pages 261–265. AAAI Press.
- André FT Martins, Marcin Junczys-Dowmunt, Fabio N Kepler, Ramón Astudillo, Chris Hokamp, and Roman Grundkiewicz. 2017. Pushing the limits of translation quality estimation. *Transactions of the Association for Computational Linguistics*, 5:205–218.
- Arindam Mitra and Chitta Baral. 2016. Addressing a question answering challenge by combining statistical methods with inductive rule learning and reasoning. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, AAAI’16*, page 2779–2785. AAAI Press.
- Rik van Noord and Johan Bos. 2017a. Dealing with co-reference in neural semantic parsing. In *Proceedings of the 2nd Workshop on Semantic Deep Learning (SemDeep-2)*, pages 41–49, Montpellier, France. Association for Computational Linguistics.
- Rik van Noord and Johan Bos. 2017b. Neural semantic parsing by character-based translation: Experiments with abstract meaning representations. *arXiv preprint arXiv:1705.09980*.
- Rik van Noord and Johan Bos. 2017c. Neural semantic parsing by character-based translation: Experiments with abstract meaning representations. *Computational Linguistics in the Netherlands Journal*, 7:93–108.

- Juri Opitz and Anette Frank. 2019. [Automatic accuracy prediction for AMR parsing](#). In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (\*SEM 2019)*, pages 212–223, Minneapolis, Minnesota. Association for Computational Linguistics.
- Juri Opitz, Letitia Parcalabescu, and Anette Frank. 2020. [Amr similarity metrics from principles](#). *Transactions of the Association for Computational Linguistics*, 8:522–538.
- Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational linguistics*, 31(1):71–106.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. [Bleu: a method for automatic evaluation of machine translation](#). In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.
- Nima Pourdamghani, Kevin Knight, and Ulf Hermjakob. 2016. [Generating English from abstract meaning representations](#). In *Proceedings of the 9th International Natural Language Generation conference*, pages 21–25, Edinburgh, UK. Association for Computational Linguistics.
- Michael Pust, Ulf Hermjakob, Kevin Knight, Daniel Marcu, and Jonathan May. 2015. [Parsing English into abstract meaning representation using syntax-based machine translation](#). In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1143–1154, Lisbon, Portugal. Association for Computational Linguistics.
- Sujith Ravi, Kevin Knight, and Radu Soricut. 2008. Automatic prediction of parser accuracy. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 887–896.
- Drew Reisinger, Rachel Rudinger, Francis Ferraro, Craig Harman, Kyle Rawlins, and Benjamin Van Durme. 2015. [Semantic proto-roles](#). *Transactions of the Association for Computational Linguistics*, 3:475–488.
- Carolina Scarton, Gustavo Paetzold, and Lucia Specia. 2016. [Quality estimation for language output applications](#). In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Tutorial Abstracts*, pages 14–17, Osaka, Japan. The COLING 2016 Organizing Committee.
- Carolina Scarton, Marcos Zampieri, Mihaela Vela, Josef van Genabith, and Lucia Specia. 2015. [Searching for context: a study on document-level labels for translation quality estimation](#). In *Proceedings of the 18th Annual Conference of the European Association for Machine Translation*, pages 121–128, Antalya, Turkey.
- Linfeng Song. 2019. [Tackling graphical NLP problems with graph recurrent networks](#). *CoRR*, abs/1907.06142.
- Linfeng Song and Daniel Gildea. 2019. [SemBleu: A robust metric for AMR parsing evaluation](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4547–4552, Florence, Italy. Association for Computational Linguistics.
- Linfeng Song, Daniel Gildea, Yue Zhang, Zhiguo Wang, and Jinsong Su. 2019. [Semantic neural machine translation using AMR](#). *Transactions of the Association for Computational Linguistics*, 7:19–31.
- Linfeng Song, Yue Zhang, Zhiguo Wang, and Daniel Gildea. 2018. [A graph-to-sequence model for AMR-to-text generation](#). In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1616–1626, Melbourne, Australia. Association for Computational Linguistics.
- Lucia Specia, Frédéric Blain, Varvara Logacheva, Ramón Astudillo, and André F. T. Martins. 2018. [Findings of the WMT 2018 shared task on quality estimation](#). In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 689–709, Belgium, Brussels. Association for Computational Linguistics.
- Lucia Specia, Nicola Cancedda, Marc Dymetman, Marco Turchi, and Nello Cristianini. 2009. Estimating the sentence-level quality of machine translation systems. In *In EAMT*, pages 28–35.
- Elias Stengel-Eskin, Aaron Steven White, Sheng Zhang, and Benjamin Van Durme. 2020. [Universal decompositional semantic parsing](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8427–8439, Online. Association for Computational Linguistics.
- Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. [Energy and policy considerations for deep learning in NLP](#). In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28–August 2, 2019, Volume 1: Long Papers*, pages 3645–3650. Association for Computational Linguistics.
- Zhenheng Tang, Yuxin Wang, Qiang Wang, and Xiaowen Chu. 2019. [The impact of gpu dvfs on the energy and performance of deep learning: An empirical study](#). In *Proceedings of the Tenth ACM International Conference on Future Energy Systems, e-Energy ’19*, pages 315–325, New York, NY, USA. ACM.

Oriol Vinyals, Łukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2015. [Grammar as a foreign language](#). In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2773–2781. Curran Associates, Inc.

Chuan Wang, Sameer Pradhan, Xiaoman Pan, Heng Ji, and Nianwen Xue. 2016. Camr at semeval-2016 task 8: An extended transition-based amr parser. In *Proceedings of the 10th international workshop on semantic evaluation (semeval-2016)*, pages 1173–1178.

Chuan Wang, Nianwen Xue, and Sameer Pradhan. 2015. [A transition-based algorithm for AMR parsing](#). In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 366–375, Denver, Colorado. Association for Computational Linguistics.

Sheng Zhang, Xutai Ma, Kevin Duh, and Benjamin Van Durme. 2019. [AMR parsing as sequence-to-graph transduction](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 80–94, Florence, Italy. Association for Computational Linguistics.

Hao Zhou, Yue Zhang, Shujian Huang, Junsheng Zhou, Xin-Yu Dai, and Jiajun Chen. 2016. [A search-based dynamic reranking model for dependency parsing](#). In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1393–1402, Berlin, Germany. Association for Computational Linguistics.

Chenxi Zhu, Xipeng Qiu, Xinchi Chen, and Xuanjing Huang. 2015. [A re-ranking model for dependency parser with recursive convolutional neural network](#). In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1159–1168, Beijing, China. Association for Computational Linguistics.

Markus Zopf. 2018. [Estimating summary quality with pairwise preferences](#). In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1687–1696, New Orleans, Louisiana. Association for Computational Linguistics.

# Generating Commonsense Explanation by Extracting Bridge Concepts from Reasoning Paths

Haozhe Ji<sup>1</sup>, Pei Ke<sup>1</sup>, Shaohan Huang<sup>2</sup>, Furu Wei<sup>2</sup>, Minlie Huang<sup>1\*</sup>

<sup>1</sup>Department of Computer Science and Technology, Institute for Artificial Intelligence,  
State Key Lab of Intelligent Technology and Systems,  
Beijing National Research Center for Information Science and Technology,  
Tsinghua University, Beijing 100084, China

<sup>2</sup>Microsoft Research

{jhz20, kp17}@mails.tsinghua.edu.cn, {shaohan, fuwei}@microsoft.com,  
aihuang@tsinghua.edu.cn

## Abstract

Commonsense explanation generation aims to empower the machine’s sense-making capability by generating plausible explanations to statements against commonsense. While this task is easy to human, the machine still struggles to generate reasonable and informative explanations. In this work, we propose a method that first extracts the underlying concepts which are served as *bridges* in the reasoning chain and then integrates these concepts to generate the final explanation. To facilitate the reasoning process, we utilize external commonsense knowledge to build the connection between a statement and the bridge concepts by extracting and pruning multi-hop paths to build a subgraph. We design a bridge concept extraction model that first scores the triples, routes the paths in the subgraph, and further selects bridge concepts with weak supervision at both the triple level and the concept level. We conduct experiments on the commonsense explanation generation task and our model outperforms the state-of-the-art baselines in both automatic and human evaluation.<sup>1</sup>

## 1 Introduction

Machine commonsense reasoning has been widely acknowledged as a crucial component of artificial intelligence and a considerable amount of work has been dedicated to evaluate this ability from various aspects in natural language processing (Levesque et al., 2011; Talmor et al., 2018; Sap et al., 2019). A large proportion of existing tasks frame commonsense reasoning as multi-choice reading comprehension problems, which lack direct assessment to machine commonsense (Wang et al., 2019) and impede its practicability to realistic scenarios (Lin

---

**Statement:** The *school* was open for *summer*.

**Explanation:** *Summertime* is typically *vacation* time for *school*.

---

Figure 1: Generating a reasonable and informative explanation involves generating *bridge concepts* like *vacation* by identifying the relation to the *source concepts*, i.e. *school* and *summer* in the statement.

et al., 2019b). Recently, Wang et al. (2019) proposed a commonsense explanation generation challenge that directly tests machine’s sense-making capability via commonsense reasoning. In this paper, we focus on the challenging explanation generation task where the goal is to generate a sentence to explain the reasons why the input statement is against commonsense, as shown in Figure 1.

Generating a reasonable explanation for a statement faces two main challenges: 1) **Trivial and uninformative explanations.** As this task can be formulated as a sequence-to-sequence generation task, existing neural language generation models tend to generate trivial and uninformative explanations. For example, one of the existing neural models generates an explanation “*The school wasn’t open for summer*” to the statement in Figure 1. Although it is sometimes reasonable, simple modification of the statement to the negation form with no additional information cannot explain the reasons why the statement conflicts with commonsense. 2) **Noisy commonsense knowledge grounding.** It’s still challenging for most existing language generation models to generate explanations that are faithful to commonsense (Lin et al., 2019b). Thus, explicitly incorporating external knowledge sources is necessary for this task. Since the nature of the explanation generation task involves using underlying commonsense knowledge to explain, locating useful commonsense knowledge from large-scale knowledge graph is not trivial and generally requires multi-hop reasoning.

\* Corresponding author

<sup>1</sup>The source code is available at <https://github.com/cdjhzh/CommExpGen>.

To address the above challenges, we propose a two-stage generation framework that first extracts the critical concepts served as *bridges* between the statement and the explanation from an external commonsense knowledge graph, and then generates plausible explanations with these concepts. We first retrieve multi-hop reasoning paths from ConceptNet (Speer et al., 2017) and heuristically prune the paths to maintain the coverage to plausible concepts while keeping the scale of the subgraph tractable. Before the extraction stage, we initialize the representation of each node on the subgraph by fusing both the contextual and graph information. Then, we design a bridge concept extraction model that scores triples, propagates the probabilities along multi-hop paths to the connected concepts and further extracts plausible concepts. In the second stage, we use a pre-trained language model (Radford et al., 2019) to generate the explanation by integrating both the statement and the extracted concept representations. Experimental results show that our framework outperforms knowledge-aware text generation baselines and GPT-2 (Radford et al., 2019) in both automatic and human evaluation. Particularly, our model generates explanations with more informative content and provides reasoning paths on the knowledge graph for concept extraction.

To summarize, our contributions are two-fold:

- We analyze the under-explored commonsense explanation generation task and investigate the challenges in incorporating external knowledge graph to aid the generation problem. To the best of our knowledge, this is the first work on generating explanations for counter-commonsense statements.
- We propose a two-stage generation method that first extracts the bridge concepts from reasoning paths and then generates the explanation based on these concepts. Our model outperforms state-of-the-art baselines on the commonsense explanation generation task in both automatic and human evaluation.

## 2 Related Work

### 2.1 Machine Commonsense Reasoning

Previous work on machine commonsense reasoning mainly focuses on the tasks of inference (Levesque et al., 2011), question answering (Talmor et al., 2018; Sap et al., 2019) and

knowledge base completion (Bosselut et al., 2019). While the ultimate goals of these tasks are different from ours, we argue that performing explicit commonsense reasoning is also critical to generation. A line of work (Bauer et al., 2018; Lin et al., 2019a) resorts to structured commonsense knowledge and builds graph-aware representations along with the contextualized word embeddings to tackle the commonsense question answering problem. In our work, we focus on reasoning over structured knowledge to explicitly infer discrete bridge concepts that are further used for text generation. Another line of work (Rajani et al., 2019; Khot et al., 2019) identifies the knowledge gap critical for the complete reasoning chain and fills the gap by writing general explanation or acquiring fine-grained annotations with human effort. While sharing a similar motivation, our method differs from theirs in the sense that we acquire distant supervisions for the bridge concepts to extract reasoning paths and generate plausible explanations without the need of additional human annotation.

### 2.2 Knowledge-Grounded Text Generation

Existing work that utilizes structured knowledge graphs to generate texts mainly lies in conversation generation (Zhou et al., 2018; Tuan et al., 2019; Moon et al., 2019), story generation (Guan et al., 2019) and language modeling (Ahn et al., 2016; Logan et al., 2019; Hayashi et al., 2019). Zhou et al. (2018) and Guan et al. (2019) propose to use graph attention that incorporates the information of neighbouring concepts into context representations to help generate the target sentence. Yang et al. (2019) resort to a dynamic concept memory that updates during essay generation. Guan et al. (2020) conduct post-training on knowledge triples to enhance the GPT-2 with commonsense knowledge. Since one-hop graphs of concepts in the statement have low coverage to the concepts in the explanation, merely leveraging information of individual concepts or triples is not suitable for this task. Another direction that utilizes more complex graph is to model multi-hop reasoning by performing random walk (Moon et al., 2019) on the knowledge graph or simulating a Markov process on the pre-extracted knowledge paths (Tuan et al., 2019). While in our task, we don't have access to a parallel grounded knowledge source nor the bridge concepts, which makes the problem even more challenging.

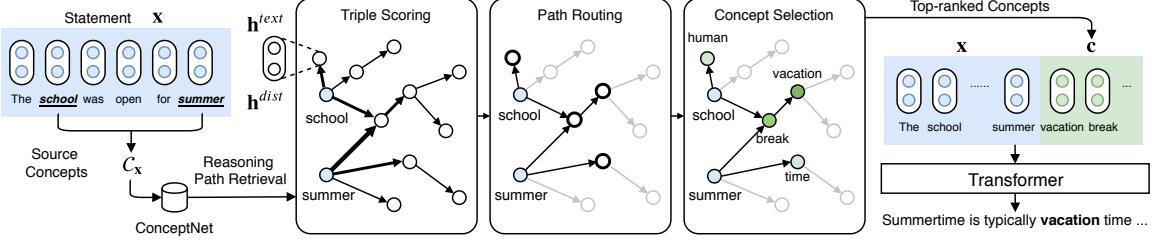


Figure 2: The inference process of our model. In the reasoning path retrieval stage (§3.3), a subgraph is firstly retrieved from the ConceptNet given the source concepts ( $C_x$ ), where each node representation is fused with both textual and graph-aware representations (§3.4). Then the model scores each triple on the subgraph, routes the path by propagating the probabilities along paths to the connected nodes, and selects concepts from activated nodes (§3.5). Finally, the model generates the explanation by integrating the token embeddings of both the statement and the top-ranked concepts (§3.6).

### 3 Methodology

#### 3.1 Task Definition

The commonsense explanation generation task is defined as generating an explanation given a statement against commonsense. Let  $\mathbf{x} = x_1 \dots x_N$  be the input statement with  $N$  words and  $\mathbf{y} = y_1 \dots y_M$  be the explanation with  $M$  words. A simple sequence-to-sequence formulation which learns a mapping from  $\mathbf{x}$  to  $\mathbf{y}$  can be adopted in this task:

$$P(\mathbf{y}|\mathbf{x}) = \prod_{t=1}^M P(y_t|\mathbf{y}_{<t}, \mathbf{x}). \quad (1)$$

#### 3.2 Model Overview

Formally, our model generates the explanation by firstly extracting the critical bridge concepts  $\mathbf{c}$  on a retrieved knowledge graph  $G_x$  given the statement  $\mathbf{x}$  and then integrating the bridge concepts and the statement to generate a proper explanation  $\mathbf{y}$ , which can be formulated as follows:

$$P(\mathbf{y}, \mathbf{c}|\mathbf{x}) = P(\mathbf{c}|\mathbf{x})P(\mathbf{y}|\mathbf{x}, \mathbf{c}) \quad (2)$$

where the bridge concepts  $\mathbf{c}$  are defined as the unique concepts delivered in the explanation but not mentioned in the statement. Figure 2 presents the overview of our model framework. Firstly, we retrieve multi-hop reasoning paths from the ConceptNet based on the statement, and heuristically prune the noisy connections to obtain a subgraph for further concept extraction (§3.3). To score the paths and concepts, we obtain the fused concept representation for each node on the subgraph by considering both the contextual and graph information (§3.4). Secondly, we design a path routing algorithm to propagate the triple probabilities along

multi-hop paths to the connected concepts and further extract plausible concepts (§3.5). Finally, our model generates the explanation by integrating the statement representation and the selected concept representation as inputs (§3.6).

#### 3.3 Reasoning Path Retrieval

In this section, we demonstrate how we retrieve and prune the reasoning paths to form a subgraph. We also acquire distant supervision for uncovering the bridge concepts in the subgraph to supervise the concept extraction in the next stage.

Given an external commonsense knowledge graph  $G = (V, E)$ , for each statement  $\mathbf{x}$ , we extract source concepts  $C_x = \{c_x^i\}$  from  $\mathbf{x}$  by aligning the surface texts in  $\mathbf{x}$  to the concepts in  $V$ . We also use the stem form of the surface texts to enable soft alignment and filter out stop words. At the training phase, we extract the target concepts  $C_y = \{c_y^j\}$  from the explanation  $\mathbf{y}$  with a similar procedure.

Starting with the source concepts, we then retrieve reasoning paths from the knowledge graph to form a subgraph that has relatively **high coverage** to the bridge concepts with a **tractable scale**.

We first examine the minimum length of paths that connect source concepts  $C_x$  with each concept in the explanation set  $C_y - C_x$ . As shown in Figure 3, over 80% of the examples require two or three hops of connection from the source concepts to the concepts that are merely mentioned in the explanation, which indicates the necessity for multi-hop reasoning.

We then count the number of concepts covered by subgraphs with different numbers of hops starting from the source concepts (We only consider concepts in the training data). As Figure 3 shows, the average number of nodes covered by 3-hop sub-

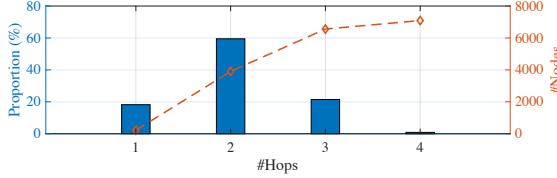


Figure 3: The left axis presents the distribution of the minimum required number of hops to reach the concepts in the explanation set  $\mathcal{C}_y - \mathcal{C}_x$  from the source concepts in  $\mathcal{C}_x$ . The right axis shows the number of nodes in the subgraph with different number of hops.

graph exceeds 6,000, indicating the need of path pruning to keep the scale tractable.

Therefore, we design a heuristic algorithm to retrieve a subgraph  $G_x = \{V_x, E_x\}$  from the ConceptNet by expanding the source concepts with 3 hops to cover most bridge concepts. To keep the scale of the subgraph tractable, at each iterating step, we enlarge  $V_x$  with  $B$  neighbour concepts most commonly visited by concepts in  $V_x$ . Intuitively, the salient bridge concepts should be in a reasonable distance from the source concepts on the graph to maintain the semantic relation and should be commonly visited nodes that support the information flow on the graph.

We distantly label the bridge concepts as the unique concepts in the explanation that could be covered by the subgraph:

$$\mathcal{B}_{x \rightarrow y} = \{c | c \in \mathcal{C}_y - \mathcal{C}_x, c \in V_x\} \quad (3)$$

### 3.4 Fused Concept Representation

We initialize each node on the subgraph with a fused concept representation  $\mathbf{h}_c$  by considering both the contextual feature of the concept and the graph-aware information. We first obtain the contextualized statement representation  $\mathbf{H}_x \in \mathbb{R}^{N \times d_1}$  using a multi-layer bi-directional Transformer encoder (Vaswani et al., 2017).

$$\mathbf{H}_x^0 = \text{one\_hot}(\mathbf{x}) \cdot \mathbf{W}_e + \mathbf{W}_p \quad (4)$$

$$\mathbf{H}_x^l = \text{trm\_block}(\mathbf{H}_x^{l-1}), l = 1, \dots, L \quad (5)$$

where  $\mathbf{W}_e$  is the token embedding matrix,  $\mathbf{W}_p$  is the position embedding matrix,  $\text{trm\_block}(\cdot)$  is the transformer block with bi-directional attention and  $L$  is the number of Transformer blocks. We typically choose the output of the last layer  $\mathbf{H}_x^L$  as the statement representation  $\mathbf{H}_x$ .

Then we consider the following embeddings:

- **Context-aware token embedding.** In order to enhance the contextual dependency of the concept  $c$  to the statement  $\mathbf{x}$ , we utilize a bi-attention network (Seo et al., 2016) that models the cross interaction between the concept and the statement.

$$\mathbf{H}_c^{tok} = \text{one\_hot}(c) \cdot \mathbf{W}_e \quad (6)$$

$$\mathbf{H}_c^{con} = \text{bi-attention}(\mathbf{H}_c^{tok}, \mathbf{H}_x) \quad (7)$$

Then we integrate  $\mathbf{H}_c^{tok}$  and  $\mathbf{H}_c^{con}$  by max pooling and linear transformation to obtain a fixed-length representation that encodes the textual information of the concept  $c$ :

$$\mathbf{h}_c^{text} = \text{mlp}\left(\max([\mathbf{H}_c^{tok}; \mathbf{H}_c^{con}])\right) \quad (8)$$

- **Concept distance embedding.** To encode the graph-aware structure information into the node representation, we design a concept distance embedding  $\mathbf{h}_c^{dist} \in \mathbb{R}^{d_1}$  that encodes the relative distance from concept  $c$  to the source concepts  $\mathcal{C}_x$  on the subgraph. Specifically, the concept distance for concept  $c$  is defined as the minimum length of the path that can be reached from one source concept in  $\mathcal{C}_x$ :

$$d_c = \min_{c_x \in \mathcal{C}_x} \text{Dist}(c_x, c) \quad (9)$$

The concept distance is then used as an index to look up a trainable matrix  $\mathbf{W}_d$  and obtain the  $\mathbf{h}_c^{dist} \in \mathbb{R}^{d_1}$ .

Finally, the fused concept representation  $\mathbf{h}_c$  is obtained by concatenating the context-aware token embedding and the concept distance embedding.

$$\mathbf{h}_c = [\mathbf{h}_c^{text}; \mathbf{h}_c^{dist}] \quad (10)$$

### 3.5 Bridge Concept Extraction

We describe the core component of our method in this section, which extracts the bridge concepts for further explanation generation. It first scores triples on the subgraph to downweight the noisy paths. Then it aggregates the path scores to each connected concepts by a path routing process and deactivates the nodes with low routing scores. Finally it selects top-ranked bridge concepts from the activated nodes.

### 3.5.1 Triple Scoring

Firstly, we calculate the triple scores according to the representation of triples and the input statement. For each triple  $e = (c_{e, \text{head}}, r_e, c_{e, \text{tail}})$  where  $c_{e, \text{head}}/c_{e, \text{tail}}$  indicates the head / tail concept and  $r_e$  denotes the relation, we can obtain its representation by concatenating the representations of the head concept, the relation and the tail concept:

$$\mathbf{h}_e = [\mathbf{h}_{c_{e, \text{head}}}; \mathbf{h}_{r_e}; \mathbf{h}_{c_{e, \text{tail}}}] \quad (11)$$

Both the head and the tail representations are calculated by Equation (10) and the relation representation is acquired by indexing a trainable relation embedding matrix  $\mathbf{W}_r$ . Then we use the statement representation to query each triple representation by taking the bilinear dot-product attention and calculate the selection probability for each triple:

$$\mathbf{h}_x = \text{max-pooling}(\mathbf{H}_x) \in \mathbb{R}^{d_1} \quad (12)$$

$$P(e|x) = \sigma(\mathbf{h}_e \mathbf{W}_2 \mathbf{h}_x^T) \quad (13)$$

We adopt weak supervision to supervise the triple scoring process. For each concept  $c \in \mathcal{B}_{x \rightarrow y}$ , we obtain the set of the shortest paths  $\mathbf{P}_{x \rightarrow c}$  using the breadth-first search from each concept of  $\mathcal{C}_x$  to  $c$ . We consider all these shortest paths  $\mathbf{P}_{x \rightarrow y} = \bigcup_{c \in \mathcal{B}_{x \rightarrow y}} \mathbf{P}_{x \rightarrow c}$  as the supervision of our triple scoring process as they connect the reasoning chain from the statement to the explanation with minimum distractive information. Accordingly, other triples in  $G_x$  which don't belong to  $\mathbf{P}_{x \rightarrow y}$  are regarded as negative samples. The loss function of triple scoring is devised as follows:

$$\begin{aligned} \mathcal{L}_{\text{triple}} = & - \sum_{e \in G_x} \mathbb{I}(e \in \mathbf{P}_{x \rightarrow y}) \log P(e|x) \\ & + [1 - \mathbb{I}(e \in \mathbf{P}_{x \rightarrow y})] \log [1 - P(e|x)] \end{aligned} \quad (14)$$

where  $\mathbb{I}(e \in \mathbf{P}_{x \rightarrow y})$  is an indicator function that takes the value 1 iff  $e \in \mathbf{P}_{x \rightarrow y}$ , and 0 otherwise.

### 3.5.2 Path Routing

Next, we describe the path routing process which involves propagating the scores along the paths to each concept on the subgraph from the source concepts. For each path  $\mathbf{p}$  retrieved from the subgraph  $G_x$ , we calculate a path score  $s(\mathbf{p})$  by aggregating the triple score  $P(e|x)$  along the path:

$$s(\mathbf{p}) = \frac{1}{|\mathbf{p}|} \sum_{e \in \mathbf{p}} P(e|x) \quad (15)$$

For each concept  $c$ , we consider all the shortest paths  $\mathbf{P}_{x \rightarrow c}$  that starts with the source concepts and ends with  $c$  monotonically, i.e., the concept distance of each node on the path increases monotonically along the path. Then we calculate the routing score for the concept  $c$  by averaging the path scores of  $\mathbf{P}_{x \rightarrow c}$ .

$$s(c) = \frac{1}{|\mathbf{P}_{x \rightarrow c}|} \sum_{\mathbf{p} \in \mathbf{P}_{x \rightarrow c}} s(\mathbf{p}) \quad (16)$$

Intuitively, this process disseminates the triple scores and aggregates them to the connected concepts. Then we deactivate some paths based on the path routing results and obtain  $V_{x \rightarrow y}$  by preserving concepts with the top- $K_1$  routing scores.

### 3.5.3 Concept Selection

Finally, we conduct concept selection based on the concept representation and the statement representation. For each concept in  $V_{x \rightarrow y}$ , we calculate the selection probability for it by taking the dot-product attention and adopt a similar cross-entropy loss with supervision from bridge concepts  $\mathcal{B}_{x \rightarrow y}$ :

$$P(c|x) = \sigma(\mathbf{h}_c \mathbf{W}_3 \mathbf{h}_x^T) \quad (17)$$

$$\begin{aligned} \mathcal{L}_{\text{concept}} = & - \sum_{c \in V_{x \rightarrow y}} \mathbb{I}(c \in \mathcal{B}_{x \rightarrow y}) \log P(c|x) \\ & + [1 - \mathbb{I}(c \in \mathcal{B}_{x \rightarrow y})] \log [1 - P(c|x)] \end{aligned} \quad (18)$$

where the indicator function is similar to that of Equation (14).

Finally, the bridge concepts with top- $K_2$  probability  $P(c|x)$  are selected as the additional input to the generation model.

## 3.6 Explanation Generation

We utilize a pre-trained Transformer decoder (Radford et al., 2019) as our generation model which shares the parameter with the Transformer encoder. Essentially, it takes the statement  $x$  and the concepts  $\mathbf{c}$  as input and auto-regressively generates the explanation  $y$ :

$$\begin{aligned} P(y|x, \mathbf{c}) &= P(y|x, c_1, \dots, c_{K_2}) \\ &= \prod_{t=1}^{M} P(y_t|x, c_1, \dots, c_{K_2}, y_{<t}) \end{aligned} \quad (19)$$

$$\mathcal{L}_{\text{generation}} = -\log P(y|x, c_1, \dots, c_{K_2}) \quad (20)$$

As shown in Figure 2, the input to the Transformer decoder is the token embeddings of both the statement and the selected concepts concatenated along the sequence length dimension.

To model bi-directional attention on the input side while preserving the causal dependency of the generated sequence, we adopt a hybrid attention mask where each token on the input side could attend to all the tokens in the input sequence while the generated token at each time step only attends to the input sequence and the previously generated tokens.

### 3.7 Training and Inference

To train the model, we optimize the final loss function which is the sum of the three loss functions:

$$\mathcal{L}_{final} = \mathcal{L}_{generation} + \lambda_1 \mathcal{L}_{triple} + \lambda_2 \mathcal{L}_{concept} \quad (21)$$

As for the inference process, Figure 2 demonstrates how our model retrieves reasoning paths given the statement, extracts bridge concepts and finally generates the explanation.

## 4 Experiment

### 4.1 Dataset and Experimental Setup

#### 4.1.1 Commonsense Explanation Dataset

We adopt the dataset from the Commonsense Validation and Explanation Challenge<sup>2</sup> which consists of three subtasks, i.e., commonsense validation, commonsense explanation selection and commonsense explanation generation. We focus on the explanation generation subtask in this paper. The commonsense explanation generation subtask contains 10,000 statements that are against commonsense. For each statement, three human-written explanations are provided. To evaluate our proposed model and other baselines, we randomly split 10% data as the test set, 5% as the development set and the latter as the training set. Note that we further split each example in the training set into three statement-explanation pairs, while for the development set and the test set we use the three corresponding explanations as references for each statement. This results in our final data split (25,596 / 476 / 992) denoted as (train / dev / test).

<sup>2</sup><https://competitions.codalab.org/competitions/21080>

#### 4.1.2 Commonsense Knowledge Graph

We use the English version ConceptNet as our external commonsense knowledge graph. It contains triples in the form of  $(h, r, t)$  where  $h$  and  $t$  represent head and tail concepts and  $r$  is the relation type. We follow Lin et al. (2019a) to merge the original 42 relation types into 17 types. We additionally define 17 reverse types corresponding to the original 17 relation types to distinguish the direction of the triples on the graph.

## 4.2 Automatic Evaluation Metrics

To automatically evaluate the performance of the generation models, we use the **BLEU-3/4** (Papineni et al., 2001), **ROUGE-2/L** (Lin, 2004), **METEOR** (Banerjee and Lavie, 2005) as our main metrics. We also propose **Concept F1** to evaluate the accuracy of the unique concepts in the generated explanation that do not occur in the statement.

Specifically, given the generated explanation  $\hat{y}$  and the reference explanation  $y$ , we extract a set of concepts  $\mathcal{C}_{\hat{y}}$  and  $\mathcal{C}_y$  from the generated explanation and the reference explanation respectively using the method in §3.3. We denote the sets of unique concepts in the explanation as  $\mathcal{U}_y = \mathcal{C}_y - \mathcal{C}_x$  and  $\mathcal{U}_{\hat{y}} = \mathcal{C}_{\hat{y}} - \mathcal{C}_x$ . Then we can compute the Concept F1 as the harmonic mean of *recall* and *precision*.

$$recall = \frac{|\mathcal{U}_{\hat{y}} \cap \mathcal{U}_y|}{|\mathcal{U}_y|}, precision = \frac{|\mathcal{U}_{\hat{y}} \cap \mathcal{U}_y|}{|\mathcal{U}_{\hat{y}}|} \quad (22)$$

## 4.3 Implementation Details

For the reasoning path retrieval process, we set the maximum number of neighbours  $B = 300$  at each hop. For each example, we restrict the concepts of the subgraph to those only appeared in the training and development set.

We use a pre-trained Transformer language model GPT-2 (Radford et al., 2019) as the initialization of the Transformer model. We set the hidden dimension  $d_1 = 768$  identical to the hidden size of the Transformer. We empirically set the following hyperparameters by tuning the model on the development set: selection threshold  $K_1 = 30, K_2 = 3$ , loss coefficients  $\lambda_1 = 1, \lambda_2 = 1$ , number of epochs = 3, batch size = 4, learning rate =  $4 \times 10^{-5}$  and use the Adam optimizer (Kingma and Ba, 2015) with 10% warmup steps. We select the model with the highest BLEU-4 score on the development set and evaluate it on the test set. At the decoding

Model	B-3/4	R-2/L	M	Concept F1
Seq2Seq	10.7/6.1	9.9/25.8	11.4	11.1
MemNet	10.2/5.7	8.8/25.7	11.0	11.5
Transformer	10.0/5.8	9.6/26.0	12.0	11.7
GPT-2-FT	23.4/15.7	18.9/36.5	17.7	17.4
Ours	<b>24.7/17.1</b>	<b>20.2/37.9</b>	<b>18.3</b>	<b>20.1</b>

Table 1: Automatic evaluation of explanation generation in terms of BLEU (B), ROUGE (R), METEOR (M) and Concept F1.

Setting	BLEU-4	Concept F1
Ours	17.1	20.1
w/o Context Emb.	16.0	18.6
w/o Distance Emb.	16.4	18.5
w/o Path Routing	16.5	19.2
#Hop = 2	16.2	18.3
#Hop = 1	15.9	17.3

Table 2: Ablation study of our framework on the test set. We present the model ablation results in the upper block and the data ablation results in the lower block.

phase, we use beam search with a beam size of 3 for all models.

#### 4.4 Baseline Models

We compare with the following baseline models:

- **Seq2Seq**: a sequence-to-sequence model based on gated recurrent unit (GRU) (Cho et al., 2014) and attention mechanism, which is widely used in text generation tasks (Bahdanau et al., 2015).
- **MemNet**: a knowledge-grounded sequence-to-sequence model (Ghazvininejad et al., 2018). In our experimental setting, we regard all the concepts which are connected with those in the statements as knowledge facts.
- **Transformer**: an encoder-decoder framework commonly used in machine translation tasks (Vaswani et al., 2017).
- **GPT-2**: a multi-layer Transformer decoder pre-trained on WebText (Radford et al., 2019) which is then directly fine-tuned on our dataset.

#### 4.5 Experimental Results

As shown in Table 1, our model achieves the best performance in terms of all the automatic evaluation metrics, which demonstrates that our model

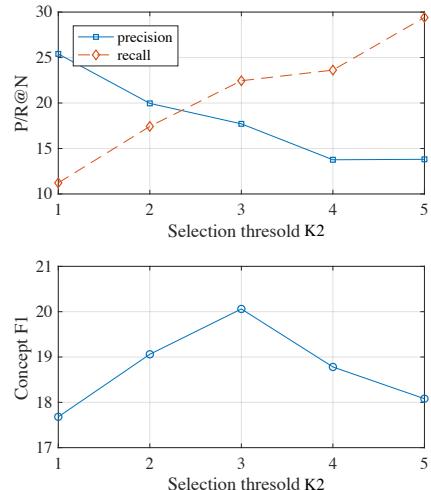


Figure 4: P/R@N measures the precision / recall of the top- $N$  selected bridge concepts. Concept F1 measures the F1-score of concepts in the generated explanations.

can generate high quality explanations. Specifically, our model achieves a 2.7% gain on Concept F1 compared with GPT-2 which indicates that explicitly extracting bridge concepts enhances the informativeness of the generated explanation.

To evaluate the effects of different modules in our method, we conduct ablation studies on both the model components and the external knowledge base. For the model components, we test the following variants: (1) without the context-aware token embeddings (**w/o Context Emb.**); (2) without the concept distance embeddings (**w/o Distance Emb.**); (3) without the path routing process (**w/o Path Routing**). As for the data ablation, we sample subgraphs by restricting the maximum number of hops to 2 (#Hop=2) and 1 (#Hop=1).

As shown in Table 2, each module contributes to the final results. Particularly, discarding the context-aware embeddings leads to the most remarkable performance drop, which indicates the significance for context modeling in multi-hop reasoning. Besides, the data ablation results demonstrate that as the subgraph has less coverage, the generation model will suffer from the noisy concepts and thus deteriorate the generation results.

We additionally present the results of the selected and generated concepts with different concepts selection threshold  $K_2$ . As shown in the upper part of the Figure 4, as the number of selected concepts increases, more true positives are selected, resulting in the increase of the recall (Recall@N) while the inclusion of more false positives leads to

Error Type	Ratio (%)	Input	Output
<b>Repetition</b>	7.7	She begins working for relaxation.	People work to <u>relax</u> , not <u>relax</u> .
<b>Overstatement</b>	19.2	Less people seek knowledge.	People <u>don't</u> seek knowledge.
<b>Unrelated</b>	26.9	The simplest carbohydrates are amino acid.	Alkaloids are not found in <u>bread</u> .
<b>Chaotic</b>	11.5	Giving assistance is for revenge.	If you help someone, you are <u>grateful</u> .

Table 3: Distribution and typical cases of different error types of the explanations generated by our model. Underlined texts denote the error types including repetition, overstatement, unrelated words and chaotic expression.

Model	Fluency		Reasonability		Informativeness	
	Win	Lose	Win	Lose	Win	Lose
vs. Seq2seq	0.41	0.02	0.86	0.04	0.84	0.05
vs. MemNet	0.48	0.00	0.84	0.03	0.87	0.03
vs. Transformer	0.33	0.01	0.71	0.03	0.72	0.03
vs. GPT-2	0.20	0.10	0.40	0.27	0.34	0.15

Table 4: Human evaluation results. The scores are the percentages of *win* and *lose* of our model in pair-wise comparison (*tie* can be calculated by  $1 - \text{win} - \text{lose}$ ). Our model is significantly better (sign test, p-value < 0.005) than all the baseline models on all three criteria.

the decrease of the precision (Precision@N). The Concept F1 reaches maximum when  $K_2 = 3$  (see the lower part), which demonstrates that the model learns to extract critical concepts for explanation generation while keeping out most noisy candidates with an appropriate selection threshold.

#### 4.6 Human Evaluation

To further evaluate the quality of the generated explanations, we conduct the human evaluation and recruit five annotators to perform pair-wise comparisons. Each annotator is given 100 paired explanations (one generated by our model and the other by a baseline model, along with the statement) and is required to give a preference among “win”, “tie”, and “lose” according to three criteria: (1) *Fluency* which measures the grammatical correctness and the readability of the explanation. (2) *Reasonability* which measures whether the explanation is reasonable and accords with the commonsense. (3) *Informativeness* which measures the amount of new information delivered in the explanation that helps explain the statement.

The results are shown in Table 4, our model outperforms all the baseline models significantly on all three criteria (sign test, p-value < 0.005). Specifically, our model wins GPT-2 substantially in terms of reasonability and informativeness.

To evaluate the inter-rater agreement for each criterion, we calculate the Fleiss’ kappa (Fleiss, 1971). For *Reasonability* / *Informativeness*, the kappa is 0.429 / 0.433 respectively indicating a

<b>Statement 1:</b> I <b>buy</b> popcorn and knife when I go to the <b>cinema</b> .
<b>Seq2Seq:</b> A person cannot <b>buy a person</b> to watch a movie.
<b>MemNet:</b> A <b>toothbrush</b> is not a place to play a movie.
<b>Transformer:</b> A <b>fridge</b> is not a place to store <b>groceries</b> .
<b>GPT-2:</b> You don't buy popcorn and knife at the cinema.
<b>Ours:</b> Knives are not <b>sold</b> at the cinema.
<b>Top-3 reasoning paths:</b> (buy → <i>antonym</i> → sell), (popcorn → <i>related to</i> → food), (cinema → <i>related to</i> → movie)
<b>Selected concepts:</b> <b>sell</b> , place, movie
<b>Statement 2:</b> He <b>eats</b> his chips with <b>toothpaste</b> .
<b>Seq2Seq:</b> <b>Chopsticks</b> are not edible.
<b>MemNet:</b> A <b>potato</b> is too soft to eat <b>juice</b> with your teeth.
<b>Transformer:</b> You do not eat <b>sand with a cup</b> .
<b>GPT-2:</b> Toothpaste is not edible.
<b>Ours:</b> Toothpaste is <b>used</b> to clean <b>teeth</b> .
<b>Top-3 reasoning paths:</b> (eat → <i>related to</i> → tooth), (toothpaste → <i>related to</i> → paste → <i>related to</i> → use), (eat → <i>has subevent</i> → work → <i>related to</i> → use)
<b>Selected concepts:</b> <b>use</b> , <b>tooth</b> , food

Table 5: Examples of generated explanations. Irrelevant contents are in red and critical concepts for explanation are in green.

moderate agreement among annotators. In terms of *Fluency*, annotators show diverse preferences ( $\kappa = 0.245$ ) since GPT-2 has strong ability in generating fluent texts.

#### 4.7 Case Study

Table 5 presents the generated explanations. Our model is capable to generate reasonable and informative explanations by utilizing the extracted bridge concepts. Specifically, in the first case our model extracts bridge concepts “sell” and identifies the incompatibility between “knives” and “cinema”. In the second case, our model clarifies the function of the “toothpaste” by extracting “use” from two reasoning paths and provides more information rather than simply negative phrasing.

## 4.8 Error Analysis

To analyze the error types of the explanations generated by our model, we manually check all the failed cases<sup>3</sup> in the pair-wise comparison between our model and the strong baseline GPT-2. The number of these cases is 26 in all 100 explanations. We manually annotated four types of errors from the failed explanations: **repetition** (words repeating), **overstatement** (overstate the points), **unrelated** concepts towards the statement (the explanation itself may be reasonable), **chaotic** sentences (difficult to understand). As shown in Table 3, it is still challenging for the model to generate explanations highly related to the statement with accurate wording.

## 5 Conclusion

In this paper, we analyze the challenges in incorporating external knowledge graph to aid the commonsense generation problem and propose a two-stage method that first extracts bridge concepts from a retrieved subgraph and then generates the explanation by integrating the extracted concepts. Experimental results show that our model outperforms baselines including the strong pre-trained language model GPT-2 in both automatic and manual evaluation.

## Acknowledgments

This work was jointly supported by the NSFC projects (key project with No. 61936010 and regular project with No. 61876096), and the Guoqiang Institute of Tsinghua University with Grant No. 2019GQG1. We thank THUNUS NExT Joint-Lab for the support.

## References

- Sungjin Ahn, Heeyoul Choi, Tanel Pärnamaa, and Yoshua Bengio. 2016. A neural knowledge language model. *ArXiv*.
- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.
- Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *IEEvaluation@ACL*.
- Lisa Bauer, Yicheng Wang, and Mohit Bansal. 2018. Commonsense for generative multi-hop question answering tasks. In *EMNLP*.
- Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. Comet: Commonsense transformers for automatic knowledge graph construction. In *ACL*.
- Kyunghyun Cho, Bart van Merriënboer, Çağlar Gülcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *EMNLP*, pages 1724–1734.
- Joseph L. Fleiss. 1971. Measuring nominal scale agreement among many raters.
- Marjan Ghazvininejad, Chris Brockett, Ming-Wei Chang, Bill Dolan, Jianfeng Gao, Wen-tau Yih, and Michel Galley. 2018. A knowledge-grounded neural conversation model. In *AAAI*.
- Jian Guan, Fei Huang, Minlie Huang, Zhihao Zhao, and Xiaoyan Zhu. 2020. A knowledge-enhanced pretraining model for commonsense story generation. *Trans. Assoc. Comput. Linguistics*, 8:93–108.
- Jian Guan, Yansen Wang, and Minlie Huang. 2019. Story ending generation with incremental encoding and commonsense knowledge. In *AAAI*.
- Hiroaki Hayashi, Zecong Hu, Chenyan Xiong, and Graham Neubig. 2019. Latent relation language models. *ArXiv*.
- Tushar Khot, Ashish Sabharwal, and Peter Clark. 2019. What’s missing: A knowledge gap guided approach for multi-hop question answering. In *EMNLP*.
- Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.
- Hector J. Levesque, Ernest Davis, and Leora Morgenstern. 2011. The winograd schema challenge. In *KR*.
- Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019a. Kagnet: Knowledge-aware graph networks for commonsense reasoning. In *EMNLP/IJCNLP*.
- Bill Yuchen Lin, Ming Shen, Yu Xing, Pei Zhou, and Xiang Ren. 2019b. Commongen: A constrained text generation dataset towards generative commonsense reasoning. *ArXiv*.
- Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *ACL*.
- Robert L. Logan, Nelson F. Liu, Matthew E. Peters, Matt Gardner, and Sameer Singh. 2019. Barack’s wife hillary: Using knowledge graphs for fact-aware language modeling. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 31, 2019*, pages 333–342. Association for Computational Linguistics.

<sup>3</sup>The decision is based on majority voting by the five annotators.

2, 2019, Volume 1: Long Papers, pages 5962–5971.  
Association for Computational Linguistics.

Seungwhan Moon, Pararth Shah, Anuj Kumar, and Rajen Subba. 2019. OpenDialKG: Explainable conversational reasoning with attention-based walks over knowledge graphs. In *ACL*, pages 845–854.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2001. Bleu: a method for automatic evaluation of machine translation. In *ACL*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Explain yourself! leveraging language models for commonsense reasoning. In *ACL*.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019. Social iqa: Commonsense reasoning about social interactions. In *EMNLP*.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *ArXiv*.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *AAAI*, pages 4444–4451.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2018. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *NAACL-HLT*.

Yi-Lin Tuan, Yun-Nung Chen, and Hung-yi Lee. 2019. DyKgChat: Benchmarking dialogue generation grounding on dynamic knowledge graphs. In *EMNLP-IJCNLP*, pages 1855–1865.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.

Cunxiang Wang, Shuaileong Liang, Yue Zhang, Xiaonan Li, and Tian Gao. 2019. Does it make sense? and why? a pilot study for sense making and explanation. In *ACL*.

Pengcheng Yang, Lei Li, Fuli Luo, Tianyu Liu, and Xu Sun. 2019. Enhancing topic-to-essay generation with external commonsense knowledge. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2002–2012, Florence, Italy. Association for Computational Linguistics.

Hao Zhou, Tom Young, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. 2018. Commonsense knowledge aware conversation generation with graph attention. In *IJCAI*, pages 4623–4629.

# Unsupervised KB-to-Text Generation with Auxiliary Triple Extraction using Dual Learning\*

Zihao Fu<sup>1</sup>, Bei Shi<sup>2</sup>, Lidong Bing<sup>3</sup>, Wai Lam<sup>1</sup>

<sup>1</sup>Department of Systems Engineering and Engineering Management,

The Chinese University of Hong Kong

<sup>2</sup>Tencent AI Lab <sup>3</sup>DAMO Academy, Alibaba Group

zhfu@se.cuhk.edu.hk; beishi@tencent.com;

l.bing@alibaba-inc.com; wlam@se.cuhk.edu.hk

## Abstract

The KB-to-text task aims at generating texts based on the given KB triples. Traditional methods usually map KB triples to sentences via a supervised seq-to-seq model. However, existing annotated datasets are very limited and human labeling is very expensive. In this paper, we propose a method which trains the generation model in a completely unsupervised way with unaligned raw text data and KB triples. Our method exploits a novel dual training framework which leverages the inverse relationship between the KB-to-text generation task and an auxiliary triple extraction task. In our architecture, we reconstruct KB triples or texts via a closed-loop framework via linking a generator and an extractor. Therefore the loss function that accounts for the reconstruction error of KB triples and texts can be used to train the generator and extractor. To resolve the cold start problem in training, we propose a method using a pseudo data generator which generates pseudo texts and KB triples for learning an initial model. To resolve the multiple-triple problem, we design an allocated reinforcement learning component to optimize the reconstruction loss. The experimental results demonstrate that our model can outperform other unsupervised generation methods and close to the bound of supervised methods.

## 1 Introduction

Knowledge Base (KB)-to-text task focuses on generating plain text descriptions from given knowledge bases (KB) triples which makes them accessible to users. For instance, given a KB triple <101 Helena, discoverer, James Craig Watson>, it is expected to generate a description sentence such as

\* The work described in this paper is substantially supported by a grant from the Research Grant Council of the Hong Kong Special Administrative Region, China (Project Codes: 14204418).

“101 Helena is discovered by James Craig Watson.”. Recently, many research works have been proposed for this task. For example, Gardent et al. (2017a,b) create the WebNLG dataset to generate description for triples sampled from DBpedia (Auer et al., 2007). Lebret et al.’s (2016) method generates people’s biographies from extracted Wikipedia infobox. Novikova et al. (2017) propose to generate restaurant reviews by some given attributes and Fu et al. (2020a) create the WikiEvent dataset to generate text based on an event chain. However, the works mentioned above usually map structured triples to text via a supervised seq-to-seq (Sutskever et al., 2014) model, in which large amounts of annotated data is necessary and the annotation is very expensive and time-consuming.

We aim to tackle the problem of completely unsupervised KB-to-text generation which only requires a text corpus and a KB corpus and does not assume any alignment between them. We propose a dual learning framework based on the inverse relationship between the KB-to-text generation task and the triple extraction task. Specifically, the KB-to-text task generates sentences from structured triples while the task of triple extraction extracts multiple triples from plain texts. Such a relationship enables the design of a closed-loop learning framework in which we link KB-to-text generation and its dual task of triple extraction so as to reconstruct the unaligned KB triples and texts. The non-differentiability issue of picking words from our neural model before reconstruction makes it hard to train the extractor or generator effectively using backpropagation. To solve this issue, we apply Reinforcement Learning (RL) based on policy gradients into our dual learning framework to optimize our extractor or generator according to the rewards.

Some semi-supervised works (He et al., 2016; Cao et al., 2019) have been proposed to generate

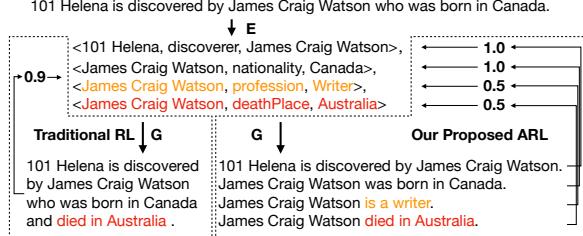


Figure 1: Illustration of the multiple-triple problem, in which  $E$  and  $G$  are extractor and generator respectively. The left part is the traditional RL methods and the right is our proposed ARL method. Four triples are extracted by the extractor. The top two triples are right and the others are wrong. Traditional RL methods give a single reward (0.9) for all the four triples while our proposed ARL gives each triple a different reward. Then the right triples and the wrong triples will be distinguished and optimized differently.

plain texts from data of certain forms in other domains (e.g., translation, semantic parsing) with limited annotated resources. These models contain two major steps. Firstly, they pre-train a weak model based on the labeled data. Secondly, they use an iterative model whose aim is to improve the weak model using the unlabeled data. In each iteration, the input sequence of the original data form is transformed into another form by the original model. Then, it is transformed back to the original data form by an inverse model. However, there are still some challenges applying the existing methods into KB-to-text directly: (1) **Cold start problem**. Existing approaches pre-train the model with labeled data and then fine-tune their models via unlabelled data. Such a mechanism still needs annotated data which is more difficult and expensive to obtain in KB-to-text task. (2) **Multiple-triple problem**. As shown in Fig. 1, multiple triples might be extracted from a text example, and inevitably, the neural extractor could extract some wrong triples. The traditional dual learning approaches (He et al., 2016; Cao et al., 2019), if directly applied, will regard all these triples as one unit and calculate a single reward for all the triples regardless of whether they are correct or not. It not only results in the slow convergence of RL, but also leads to unsatisfactory model performance.

We propose a novel Extractor-Generator Dual (EGD) framework which exploits the inverse relationship between KB-to-text generation and auxiliary triple extraction. Our model can resolve the KB-to-text task in a totally unsupervised way. To cope with the cold start problem, we propose a

pseudo data generator (PDG) which can generate pseudo text and pseudo KB triples based on the given unaligned KB triples and text respectively with prior knowledge. The extractor and the generator are then pre-trained with the generated pseudo data. To resolve the multiple-triple problem, we propose a novel Allocated Reinforcement Learning (ARL) component. Different from traditional RL methods in which one reward is calculated for the whole sequence, ARL allocates different rewards to different sub-parts of the sequence (Fig. 1 right). Therefore, our model can distinguish the quality of each triple and optimize the extractor and the generator more accurately. We compare our framework with existing dual learning methods and the experimental results demonstrate that our model can outperform other unsupervised generation methods and close to the bound of supervised methods.

## 2 Related Works

Recently many tasks and methods have been proposed to transform existing data into human-readable text. WebNLG (Gardent et al., 2017a,b) is proposed to describe a list of triples sampled from DBpedia (Auer et al., 2007). Except for the KB triples, many other types of data have also been investigated for how to generate text from them. For example, E2E (Novikova et al., 2017) aims at generating text from some restaurants’ attributes. Wikibio (Lebret et al., 2016) proposes to generate biographies for the Wikipedia infobox while WikiEvent (Fu et al., 2020a) proposes to generate text based on an event chain. Besides, Chen and Mooney (2008); Wiseman et al. (2017) propose to generate a summarization of a match based on the scores and Liang et al. (2009) propose to generate weather reports based on the records. All these tasks require an elaborately annotated dataset which is very expensive to prepare.

Many methods have been proposed to tackle the dataset insufficiency problem in other tasks. Fu et al. (2020c) propose to directly train the model on partially-aligned data in which the data and the text are not necessarily exactly math, and it can be built automatically. He et al. (2016); Sennrich et al. (2016); Yi et al. (2017) propose dual learning frameworks. They pre-train a weak model with parallel data and refine the model with monolingual data. This strategy has been applied in many related tasks including semantic parsing (Cao et al., 2019), summarization (Baziotis et al., 2019) and

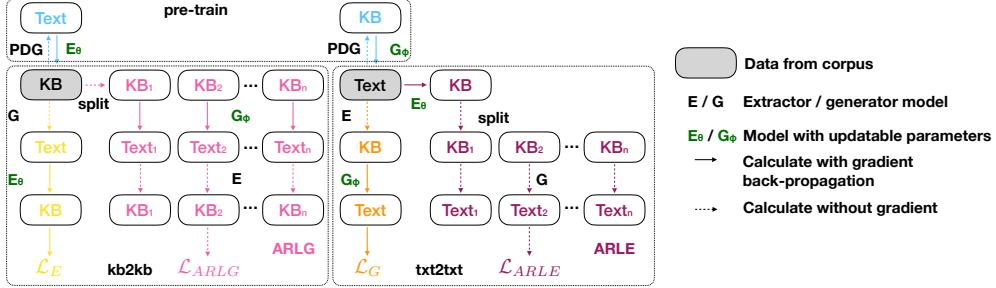


Figure 2: The extractor-generator dual (EGD) framework. It contains three processes namely a pre-train process, a kb2kb process and a txt2txt process.

information narration (Sun et al., 2018). However, as indicated in Hoang et al. (2018), the dual learning approach is not easy to train. Moreover, these methods still need some aligned data to pre-train the weak model. Another line of research proposes to use some extra annotations instead of using aligned data. Lample et al. (2018a,b) propose to train an unsupervised NMT system based on few annotated word pairs (Conneau et al., 2018). Luo et al. (2019) propose to generate pseudo data with a rule-based template (Li et al., 2018). However, these models cannot be directly applied in our scenario since our dataset is too complicated to make these annotations. Fu et al. (2020b) propose to utilize topic information from a dynamic topic tracker to solve the dataset insufficiency problem. Cheng et al. (2020) propose to generate better text description for a few entities by exploring the knowledge from KB and distill the useful part. In the field of computer vision, Zhu et al. (2017) propose cycleGAN which uses a cycled training method that transforms the input into another data form and then transforms it back, minimizing the recover loss. The method works well in the image domain but has some problems in text generation considering the non-differentiable discrete layer. We follow the ideas of cycleGAN to train the whole model without supervised data and adopt the RL method proposed in dual learning methods.

Reinforcement Learning (RL) has been utilized to solve the infeasibility of backpropagation through discrete tokens layer. Li et al. (2016) propose to use RL to focus on the long term target and thus improve the performance. Yu et al. (2017) propose to use the RL in generative adversarial networks to solve the discrete tokens problem. He et al. (2016); Sun et al. (2018) propose to use RL in dual training. As far as we know, no studies of RL have been conducted for KB triples in which

the reward is different for each triple considering multiple-triple problem.

### 3 Method

#### 3.1 Problem Definition

Formally, we denote the KB corpus as  $\mathcal{K} = \{K_i | \forall i\}$  in which  $K_i = [k_1^{(i)}, k_2^{(i)}, \dots, k_{n_i}^{(i)}]$  is the  $i$ th KB triple list containing  $n_i$  triples.  $k_j^{(i)} = (h_j^{(i)}, r_j^{(i)}, t_j^{(i)})$  represents the  $j$ th KB triple in  $K_i$  containing the head, relation and tail entity respectively. We denote the texts corpus as  $\mathcal{T} = \{T_i | \forall i\}$  in which  $T_i = [t_1^{(i)}, t_2^{(i)}, \dots, t_{n_i}^{(i)}]$  is the  $i$ th sentence and  $t_j^{(i)}$  is the  $j$ th word in the sentence. In our problem, we are only given a collection of KB triples  $\mathcal{K}_t \subset \mathcal{K}$  and a collection of text  $\mathcal{T}_t \subset \mathcal{T}$  without any alignment information between them. The ultimate goal is to train a model that generates the corresponding text in  $\mathcal{T}$  describing the given triple list from  $\mathcal{K}$ .

#### 3.2 Extractor-Generator Dual Framework

Our proposed Extractor-Generator Dual (EGD) framework is composed of a generator  $G$  and an extractor  $E$  that translate data in one form to another. We denote all trainable parameters in  $E$  and  $G$  as  $\theta$  and  $\phi$ , respectively. The generator generates text representation for each KB triple as  $T' = G(K), K \in \mathcal{K}, T' \in \mathcal{T}$  while the extractor extracts KB triples from raw text as  $K' = E(T), T \in \mathcal{T}, K' \in \mathcal{K}$ . Our EGD framework is trained in an unsupervised manner and it contains three processes, as shown in Fig. 2. The first process is a pre-train process in which both  $E$  and  $G$  are trained with the pseudo data generated by the pseudo generator. The second process is the kb2kb process which generates description text based on the given KB triples with  $G$  and then recovers the KB triples from the generated text with  $E$ . The

third process is called txt2txt which extracts KB triples from the given text with  $E$  and then recovers the text from the generated KB triples with  $G$ . In order to overcome the multiple-mapping problem, we propose a novel allocated reinforcement learning component in kb2kb and txt2txt, respectively.

The EGD framework firstly pre-trains the extractor and generator with the data generated by the pseudo data generator (PDG). For the text corpus  $\mathcal{T}_t$ , we generate corresponding pseudo KB triples as  $\mathcal{K}'_t = \{K = P_K(T) | \forall T \in \mathcal{T}_t\}$ , in which  $P_K$  is the pseudo KB generator. We pre-train the generator  $G$  to transform  $K \in \mathcal{K}'_t$  to  $T \in \mathcal{T}_t$ . Similarly, we generate pseudo text as  $\mathcal{T}'_t = \{T = P_T(K) | \forall K \in \mathcal{K}_t\}$ , in which  $P_T$  is the pseudo text generator. Then, we train the extractor to transform  $T \in \mathcal{T}'_t$  to  $K \in \mathcal{K}_t$ . After  $G$  and  $E$  have been pre-trained, the kb2kb process and the txt2txt process are conducted alternately to further improve the performance.

In the kb2kb process, the input KB triples are firstly flattened and concatenated one by one as  $K = [k_1, k_2, \dots, k_{n_k}] = [w_1, w_2, \dots, w_{n_w}]$  in which  $k_i$  is the  $i$ th triple in  $K$  while  $w_i$  denotes the  $i$ th words in the concatenated word list.  $n_k$  is the number of triples while  $n_w$  is the number of the words.  $K$  is then sent into the generator  $G$  to get a text description  $T_m = [t_1, t_2, \dots, t_{n_t}]$ , where  $t_i$  is the  $i$ th word in the sentence  $T_m$  and  $n_t$  is the length of  $T_m$ . Afterwards, The extractor takes the sentences  $T_m$  as input and outputs the triple sequence  $K' = [w'_1, w'_2, \dots, w'_{n'_w}]$ , in which  $w'_i$  is the  $i$ th word in  $K'$  while  $n'_w$  is the length of  $K'$ . The target is to make  $K'$  as close to  $K$  as possible. Therefore, in the training step, the loss function for the extractor is defined as the negative log probability of each word in  $K$ :

$$\mathcal{L}_E = - \sum_{i=1}^{n_w} \log p_\theta(w'_i = w_i | T_m, w_1, \dots, w_{i-1}).$$

We can also use the output to improve the generator. Since  $T_m$  is discrete, the gradient cannot be passed to the generator as the cycleGAN (Zhu et al., 2017) does. To tackle this problem, we propose an Allocated Reinforcement Learning for Generator (ARLG) component to utilize the extractor's result to optimize the generator. Different rewards are allocated to different parts of the generator output. The gradient for the generator is denoted as  $\nabla_\phi \mathcal{L}_{ARLG}$  which will be introduced in the later section.

In the txt2txt process, the input text  $T = [t_1, t_2, \dots, t_{n_t}]$  is transformed into its KB representation  $K_m = [k_1, k_2, \dots, k_{n_m}]$  by the extractor  $E$ .  $K_m$  is then transformed to  $T' = [t'_1, t'_2, \dots, t'_{n_t}]$  by the generator and the loss is defined as:

$$\mathcal{L}_G = - \sum_{i=1}^{n_t} \log p_\phi(t'_i = t_i | K_m, t_1, \dots, t_{i-1}).$$

Similarly, we also propose an Allocated Reinforcement Learning for Extractor (ARLE) to utilize the generator's result to optimize the extractor. Different rewards are allocated to different parts of the extractor output. Let the gradient for the extractor be denoted as  $\nabla_\theta \mathcal{L}_{ARLE}$ . The final gradient for extractor's parameters  $\theta$  is formulated as  $\nabla_\theta \mathcal{L}_E + \nabla_\theta \mathcal{L}_{ARLE}$  while the gradient for generator's parameters  $\phi$  is  $\nabla_\phi \mathcal{L}_G + \nabla_\phi \mathcal{L}_{ARLG}$ . We use the Adam (Kingma and Ba, 2014) as the optimizer to optimize all the parameters.

### 3.3 Background of Transformer

The extractor and the generator are both backboned by the prevalent Transformer (Vaswani et al., 2017) model, which is a variant of the seq-to-seq model. It takes a sequence as input and generates another sequence as output. The Transformer model contains two parts, namely an encoder and a decoder. Both of them are built with several attention layers. We refer readers to the original paper (Vaswani et al., 2017) for more details.

### 3.4 Pseudo Data Generator

To handle the cold start problem, we propose a novel pseudo data generator (PDG) to generate pseudo data. It contains two components, namely a pseudo text generator and a pseudo KB generator.

**Pseudo Text Generator** generates pseudo text for each KB and forms a pseudo supervised training data for pre-training the extractor and thus solving the cold start problem. We compute a statistics of the word count in the training set  $\mathcal{T}_t$  and calculate the empirical distribution for each word as:

$$p(w) = \frac{\#w}{\sum_{w' \in \mathcal{T}_t} \#w'},$$

where  $\#w$  stands for the total word count for  $w$  in  $\mathcal{T}_t$ . For a list of KB triples  $K = [k_1, k_2, \dots, k_{n_k}] = [h_1, r_1, t_1, h_2, r_2, t_2, \dots, h_{n_k}, r_{n_k}, t_{n_k}]$ , we firstly sample head entities and tail entities as  $K_s = [h_1, t_1, h_2, t_2, \dots, h_n, t_n]$ . The final

sequence is generated by sampling from both  $K_s$  and  $p(w)$ . When generating each word  $\tilde{T}_i$ , a random number generator is used to generate a random number  $r_i$  uniformly.  $r_i$  is used to compare with a threshold parameter  $\alpha$ . If  $r_i > \alpha$ ,  $\tilde{T}_i$  is sampled with the word distribution  $p(w)$ , otherwise, it is sampled from the next token in  $K_s$ . This process can be expressed mathematically as:

$$\tilde{T}_i = \begin{cases} w \sim p(w) & r_i > \alpha \\ K_s[1 + \sum_{j=1}^{i-1} \mathbb{1}(\tilde{T}_j \in K_s)] & \text{otherwise} \end{cases}$$

in which  $\mathbb{1}(C) = 1$  if condition  $C$  is true and 0 otherwise.  $\tilde{T}_j \in K_s$  indicates whether the word  $\tilde{T}_j$  is sampled from  $K_s$ . This pseudo text data is used to solve the cold start problem when training the extractor.

**Pseudo KB Generator** generates pseudo KB triples for each text and form a pseudo supervised training data. This data is used to solve the cold start problem when pre-training the generator. Similar with the work of Freitag and Roy (2018), for an input sequence  $T$  we randomly remove words in the input text with a probability  $\beta_1$  and sample new words by sampling words from a distribution with a probability  $\beta_2$ . The generated sequence  $\tilde{K}$  is the pseudo KB sequence for each text. Similar to the Pseudo Text Generator, we randomly add some words by sampling from the distribution  $p(w)$ . We do not use the probability calculated from  $\mathcal{K}_t$  since it may sample some wrong relations or wrong entity names which undermines the performance. Mathematically, it can be expressed as:

$$\tilde{K}_i = \begin{cases} w \sim p(w) & r_i < \beta_2 \\ T_s[1 + \sum_{j=1}^{i-1} \mathbb{1}(\tilde{K}_j \in T_s)] & \text{otherwise} \end{cases}$$

in which  $T_s = s(T)$  and  $s(\cdot)$  is a sample function defined as:

$$s(T) = \begin{cases} T & \|T\| = 0 \\ [T_1; s(T_{2:\|T\|})] & r < \beta_1, \|T\| \neq 0 \\ s(T_{2:\|T\|}) & \text{otherwise} \end{cases}$$

where  $\|T\|$  denotes the length of the sequence  $T$  while  $T_{2:\|T\|}$  stands for the sub-sequence from the second to the last of  $T$ .

### 3.5 Allocated Reinforcement Learning

Traditional reinforcement learning for sequence generation calculates a reward for the whole sequence (He et al., 2016; Hoang et al., 2018; Keshlloo et al., 2018) and uses the policy gradient (Sutton et al., 2000) algorithm to optimize the parameters. It suffers from the multiple-triple problem as discussed above. We propose an allocated reinforcement learning method to allocate different rewards for different KB triples and thus alleviate this problem. In the kb2kb process, the RL model is called the Allocated Reinforcement Learning for Generator (ARLG) since it optimizes the parameters in the generator while in the txt2txt process, it is called Allocated Reinforcement Learning for Extractor (ARLE) accordingly.

**ARLE** is shown in Fig. 2. The main idea is to recover and evaluate the KB triples separately which inherently has the following benefits: 1) Each triple is given a distinct reward as discussed above; 2) Traditional RL is more likely to ignore some triples (e.g., 3rd triple in Fig. 1) since it handles several triples at once while our method alleviates such problem by handling triples one by one. It firstly sends the input text  $T = [t_1, t_2, \dots, t_{n_t}]$  into the extractor and get the extracted triples:  $K_m = E(T) = [k_m^{(1)}, k_m^{(2)}, \dots, k_m^{(n_k)}]$ . The corresponding probability for each token is denoted as  $p_j^{(i)}$ , in which  $i$  denotes the  $i$ th triple and  $j$  denotes the  $j$ th word in the triple. Afterwards, the generator is applied on each triple in  $K_m$  to recover the corresponding text, which denotes as:  $T' = [G(k_m^{(1)}), G(k_m^{(2)}), \dots, G(k_m^{(n_k)})] = [t'_1, t'_2, \dots, t'_{n_k}]$ . We calculate the reward for each  $k_m^{(i)}$  as the recall for each corresponding  $t'_i$  referring to  $T$ :

$$R(k_m^{(i)}) = \frac{\sum_{j=1}^{\|t'_i\|} \mathbb{1}(t_i'^{(j)} \in T)}{\|t'_i\|},$$

in which  $\|t'_i\|$  denotes the length of  $t'_i$  and  $t_i'^{(j)}$  is the  $j$ th word in  $t'_i$ . The reward for each sentence in  $K_m$  is denoted as:  $R_e = [R(k_m^{(1)}), R(k_m^{(2)}), \dots, R(k_m^{(n_k)})]$ . Different from the traditional policy gradient algorithm (Sutton et al., 2000), our RL uses a different reward for each generated triple. The gradient is calculated as:

$$\nabla_{\theta} \mathcal{L}_{ARLE} = -\mathbb{E}[\sum_{i=1}^{n_k} R(k_m^{(i)}) \sum_{j=1}^{\|k_m^{(i)}\|} \nabla_{\theta} \log p_j^{(i)}].$$

Since the RL model only guides the model with some reward scores which is only one aspect of the result. It misleads the model into generating some sequences which have a high reward while actually perform worse. To prevent this, we propose to conduct the gradient descent together with the kb2kb process simultaneously in which the extractor is trained with a supervised sequence.

**ARLG** is applied in the kb2kb process. The input KB triples is firstly splitted into  $n_k$  triples  $K = [k_1, k_2, \dots, k_{n_k}]$  which is then sent into the generator separately and get the corresponding description sentences:  $T_m = [G(k_1), G(k_2), \dots, G(k_{n_k})] = [t_m^{(1)}, t_m^{(2)}, \dots, t_m^{(n_k)}]$ . The corresponding probability for the  $j$ th word in the  $i$ th sentence is denoted as  $p_j^{(i)}$ . Afterwards, the text is sent into the extractor to recover the input KB triple for each  $t_m^{(i)}$ :  $K' = [E(t_m^{(1)}), E(t_m^{(2)}), \dots, E(t_m^{(n_k)})] = [k'_1, k'_2, \dots, k'_{n_k}]$ . We calculate the reward for each  $t_m^{(i)}$  as the precision for each corresponding  $k'_i$  referring to  $k_i$  in  $K$ :

$$P(t_m^{(i)}) = \frac{\sum_{j=1}^{\|k'_i\|} \mathbb{I}(k'_i \in k_i)}{\|k'_i\|},$$

in which  $\|k'_i\|$  denotes the total word number count of  $k'_i$ . The reward for each sentence in  $T_m$  is denoted as:  $R_g = [P(t_m^{(1)}), P(t_m^{(2)}), \dots, P(t_m^{(n_k)})]$ . We use RL to maximize the expected reward for each KB triple  $t_m^{(i)}$  with corresponding reward. The gradient is:

$$\nabla_\phi \mathcal{L}_{ARLG} = -\mathbb{E} \left[ \sum_{i=1}^{n_k} P(t_m^{(i)}) \sum_{j=1}^{\|t'_i\|} \nabla_\theta \log p_j^{(i)} \right].$$

Similar to ARLE, we also train the model with the txt2txt process to give a targeted sequence to guide the training together with the reward score.

## 4 Experiments

### 4.1 Dataset

We adopt the WebNLG v2 dataset (Gardent et al., 2017a)<sup>1</sup>. It samples KB triples from DBpedia and annotates corresponding texts by crowdsourcing. In order to show that our model can work under the unsupervised setting, we split the original dataset into two parts, namely the KB part and the text part. We do not assume any alignment between

<sup>1</sup><https://gitlab.com/shimorina/webnlg-dataset>

#triples	1	2	3	4	5	6	7	Total
train	7,429	6,717	7,377	6,888	4,982	488	471	34,352
dev	924	842	919	877	632	64	58	4,316
test	931	831	903	838	608	58	55	4,224

Table 1: Statistics for the dataset. The number of instances with different number of triples are listed.

KB and text. Table 1 shows the statistics of instances with different number of triples. In this dataset, one sentence can be mapped to at most seven triples. We use the same dev and test set as the original WebNLG. The training set has 34,352 samples in total while the dev set and the test set have 4,316 and 4,224 samples respectively. It can be observed that there are 78.2% sentences mapped with multiple-triple.

### 4.2 Comparison Models

We compare our model against the following baseline methods:

**PDG** uses the Pseudo Data Generator to generate the pseudo data for pre-training both extractor and generator. PDG does not conduct the subsequent dual learning process and thus illustrates the capability of PDG.

**DL** uses the dual learning process proposed in He et al. (2016); Zhu et al. (2017). It is fine-tuned on the PDG model and iterates alternatively between txt2txt and kb2kb processes. Here, we do not use any reinforcement learning component.

**DL-RL1** uses the dual learning process together with an RL component. It is similar to the dual learning method proposed in He et al. (2016); Zhu et al. (2017). We use the PDG’s data to train the weak model. It uses the log-likelihood of the recover process’s output sequence as the reward.

**DL-RL2** follows the settings of Sun et al. (2018). Different from DL-RL1, this model uses the ROUGE<sub>L</sub> (Lin, 2004) score of the recovered sequence instead of using the log-likelihood as the reward.

**SEG** is a Supervised Extractor-Generator using the original setting of WebNLG for both generator and extractor. It utilizes all the alignment information between KB and text and thus provides an upper bound for our experiment.

### 4.3 Experimental settings

We evaluate the performances of the generator and the extractor with several metrics including BLEU (Papineni et al., 2002), NIST (Dodding-

	Generator					Extractor								
	BLEU	NIST	METEOR	ROUGE <sub>L</sub>	CIDEr	BLEU	NIST	METEOR	ROUGE <sub>L</sub>	CIDEr	Precision	Recall	F1	
PDG	0.322	7.06	0.349	0.505	2.63	0.489	6.01	0.351	0.618	3.97	0.635	0.465	0.510	
DL	0.352	7.71	0.347	0.528	2.96	0.735	10.4	0.502	0.743	5.67	0.644	0.691	0.646	
DL-RL1	0.356	7.73	0.350	0.532	3.00	0.760	10.8	0.501	0.755	5.92	0.670	0.687	0.658	
DL-RL2	0.356	7.75	0.350	0.533	2.99	0.757	10.7	<b>0.503</b>	0.755	5.90	0.668	<b>0.691</b>	0.659	
EGD	<b>0.369</b>	<b>7.77</b>	<b>0.364</b>	<b>0.541</b>	<b>3.13</b>	<b>0.775</b>	<b>11.1</b>	<b>0.503</b>	<b>0.772</b>	<b>6.25</b>	<b>0.704</b>	<b>0.691</b>	<b>0.680</b>	
EGD w/o ARLE	0.351	7.72	0.347	0.529	2.97	0.770	10.9	0.501	0.764	6.11	0.683	0.682	0.665	
EGD w/o ARLG	0.353	7.77	0.348	0.531	2.99	0.729	10.4	0.505	0.746	5.61	0.639	0.695	0.645	
EGD w/o PDG	0.010	0.82	0.037	0.119	0.02	0.020	0.42	0.026	0.042	0.08	0.011	0.008	0.007	
SEG	0.406	8.31	0.385	0.585	3.66	0.848	11.8	0.595	0.867	7.43	0.783	0.830	0.796	

Table 2: Results for generator (left) and extractor (right), which are evaluated with generation metrics. For the extractor, precision, recall, and F1 scores are also calculated at triple’s level. The performances of our EGD method without different components and the supervised method SEG are shown in the bottom.

Ratio	Generator		Extractor	
	BLEU	ROUGE <sub>L</sub>	BLEU	ROUGE <sub>L</sub>
0.10	0.235	0.439	0.335	0.557
0.15	0.281	0.49	0.655	0.708
0.20	0.308	0.506	0.746	0.757
0.25	0.347	0.524	0.71	0.764
PDG	0.322	0.505	0.489	0.618

Table 3: Compare our PDG framework with semi-supervised models at different labeling ratios.

ton, 2002), METEOR (Banerjee and Lavie, 2005), ROUGE<sub>L</sub> (Lin, 2004), and CIDEr (Vedantam et al., 2015). These metrics are calculated with the evaluation code provided in Novikova et al. (2017). Moreover, we also evaluate the performance of the extractor with precision, recall, and F1 scores (Manning et al., 2010). In PDG, we set  $\alpha = 0.8$ ,  $\beta_1 = 0.2$ ,  $\beta_2 = 0.6$ . We firstly pre-train the extractor and the generator in the PDG model with the data generated by PDG until convergence. All other models are fine-tuned on the PDG model. For the DL model, we train the generator for 5 steps with the txt2txt process and train the extractor with the kb2kb process for another 5 steps with the new generator. We iterate this process 10 times. For all transformers, we set clip norm to 1.0, label smoothing to 0.1, and dropout to 0.3. We use Adam (Kingma and Ba, 2014) as our optimizer and set the learning rate for the extractor to 2e-4 and generator to 5e-4. All hyper-parameters are tuned on the dev dataset with grid search.

#### 4.4 Experimental Results

The performances of our KB-to-text generator and triple extractor are shown in the left and right of Table 2 respectively. Both generator and extractor of our model outperform all baseline models significantly and consistently. The comparison between our EGD model and the supervised SEG model indicates that our unsupervised EGD model

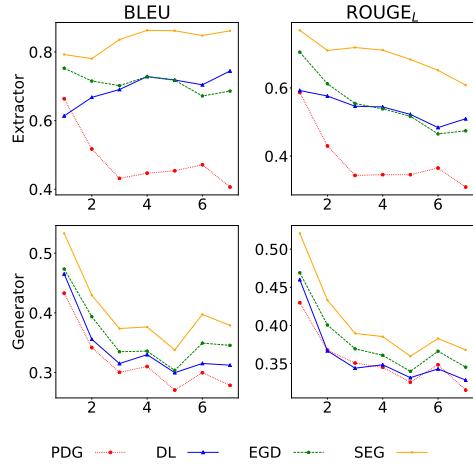


Figure 3: The influence of KB triples count. The  $x$ -axis represents the KB triples count while the  $y$ -axis represents the scores.

is close to the bound of the supervised methods. Compared with the PDG model, our EGD model has a much better performance with the dual learning framework and the ARL component. Moreover, Our EGD model outperforms the DL-RL1 and DL-RL2 model, which indicates that our proposed ARL component can handle the multiple-triple problem between triples and texts. In the traditional RL models, the reward is the same for a whole sequence including all the triples while in our ARL model, the reward is calculated for several sub-parts of the sequence, which is more accurate and effective. By comparing PDG with SEG, we found that the model trained with our proposed pseudo data generator (PDG)’s output achieves acceptable results. It indicates that using the PDG’s output is a feasible alternative to initialize the model and can handle the cold start problem.

**Ablation Study.** We also conduct some ablation studies to show that each component contributes

	Extractor	Generator
Gold	(1634 : The Ram Rebellion, mediaType, E - book) (1634 : The Ram Rebellion, author, Virginia DeMarce)	Virginia DeMarce is the author of 1634 : The Ram Rebellion , which can be found as an e - book .
SEG	(1634 : The Ram Rebellion, mediaType, E - book) (1634 : The Ram Rebellion, author, Virginia DeMarce)	1634 : The Ram Rebellion was written by Virginia DeMarce and has the ISBN number 1 - 4165 - 2060 - 0 .
PDG	(1634 : The Ram Rebellion, mediaType, E - book)	1634 : The Ram Rebellion was followed by 1634 : The Galileo Affair and its author is Virginia DeMarce .
DL	(1634 : The Ram Rebellion, mediaType, E - book) (1634 : The Ram Rebellion, author, Virginia DeMarce) (1634 : The Ram Rebellion, ISBN number, 1 - 4165 - 2060 - 0 )	1634 : The Ram Rebellion is available as an E - Book .
EGD	(1634 : The Ram Rebellion, mediaType, E - book) (1634 : The Ram Rebellion, author, Virginia DeMarce)	Virginia DeMarce is the author of 1634 : The Ram Rebellion , currently in print .

Table 4: Case study. The input KB and text are listed in the first row.

to the final performance. The results are shown at the bottom part of Table 2. By comparing the model EGD w/o ARLE and EGD w/o ARLG with the EGD model, we can see that both the ARLE and ARLG components are effective to handle the multiple-triple problem and help improve the performance. It is interesting to see that the result of EGD w/o PDG is extremely poor showing the importance of our PDG component. The EGD w/o PDG removes the pre-train stage with the pseudo data generator and conducts the iterations between txt2txt and kb2kb directly. Without PDG, we observe that the models tend to learn some “own language” without a good initialization which is incomprehensible to human.

**The Influence of the KB triples Number.** We analyze the influence of the KB triples’ number on the performance. The results are shown in Fig. 3. As expected, the SEG model performs the best over all numbers since it is fully supervised. The PDG model performs the worst since it only uses pseudo data to train. The DL model improves significantly comparing with the PDG model over all numbers, especially in the extractor model. It shows that using dual learning’s iteration approach does improve the model of training solely based on PDG’s data. Our proposed EGD model outperforms the DL model and the PDG model. This shows that the ARL model does help to give more information to train the model. Nearly all generators’ scores decrease as the number increases. This is because if the sequence is long, it has more ways to express those triples which may be different from the gold standard sentence. However, when extracting triples from the text, it only has one correct way and thus the extractor’s scores are similar in all lengths.

**Error Analysis.** We conduct an error analysis experiment for the top 20 mentioned relations in

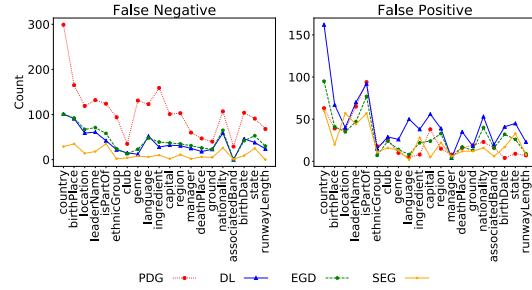


Figure 4: Error analysis for top 20 mentioned relations.

the extractor which is shown in Fig. 4. We focus on two kinds of errors. The first kind of error is called “false negative” which means when extracting, some correct triples are ignored. The second kind of error is called “false positive” which means that the extractor generates some incorrect triples that the text does not mention. It can be observed that the “false negative” problem is much more severe than the “false positive” problem for the PDG model, while the DL model and the EGD model alleviate this problem a lot. This reason is that the pseudo text data is made by sampling entities in KB ignoring relation information. Iterating alternately between txt2txt and kb2 solves the problem since the missing information is supplemented. It can also be observed that when comparing with the DL model, our EGD model mainly solves the “false positive” problem. The reason is that the RL can penalize the wrong generated triples but cannot give specific guidance on which missing triples the model should generate.

#### Comparison with Semi-Supervised Learning.

To measure the quality of the initialization via PDG, we compare our PDG method against the semi-supervised learning method. We sample labeled data from the original dataset with different ratios to train the models and compare the results with the PDG model. The result is shown in Table 3. It can be concluded from the result that training the

extractor with the PDG’s data outperforms training with 10% aligned data and it also outperforms 20% aligned data for the generator. It shows that our PDG component does provide usable data and it can be boosted a lot in the subsequent dual iteration process.

**Case Study.** Table 4 shows a case study for 4 models. For the extractor, the input is “*Virginia DeMarce is the author of 1634 : The Ram Rebellion , which can be found as an e - book .*”. For the generator, the input is “(*1634 : The Ram Rebellion, mediaType, E - book*) (*1634 : The Ram Rebellion, author, Virginia DeMarce*)”. It can be observed that for the PDG model, it omits the second triple. It also shows that the PDG model has a severe false negative problem which has been mentioned in the error analysis sub-section. The DL model alleviates this problem but it introduces more triples causing the false positive problem. Our EGD model solves the false positive problem by the RL component. All models make some mistakes in the generation process including the supervised SEG model. The result of the generator shows that it is more difficult to generate a sequence than extracting triples.

## 5 Conclusions

We propose a new challenging task, namely, unsupervised KB-to-text generation. To solve this task, we propose an extractor-generator dual framework which exploits the inverse relationship between the KB-to-text generation task and the auxiliary triple extraction task. To handle the cold start problem and the multiple-triple problem respectively, we propose a novel pseudo data generator and an allocated reinforcement learning component. Experimental results show that our proposed method successfully resolves the observed problems and outperforms all the baseline models.

## References

- Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary G. Ives. 2007. Dbpedia: A nucleus for a web of open data. In *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference*, pages 722–735.
- Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72.
- Christos Baziotis, Ion Androutsopoulos, Ioannis Konstas, and Alexandros Potamianos. 2019. Seq<sup>~3</sup>: Differentiable sequence-to-sequence-to-sequence autoencoder for unsupervised abstractive sentence compression. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 673–681.
- Ruisheng Cao, Su Zhu, Chen Liu, Jieyu Li, and Kai Yu. 2019. Semantic parsing with dual learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 51–64, Florence, Italy. Association for Computational Linguistics.
- David L Chen and Raymond J Mooney. 2008. Learning to sportscast: a test of grounded language acquisition. In *Proceedings of the 25th international conference on Machine learning*, pages 128–135. ACM.
- Liying Cheng, Dekun Wu, Lidong Bing, Yan Zhang, Zhanming Jie, Wei Lu, and Luo Si. 2020. Ent-desc: Entity description generation by exploring-knowledge graph. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.
- Alexis Conneau, Guillaume Lample, Marc’Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *International Conference on Learning Representations (ICLR)*.
- George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the Second International Conference on Human Language Technology Research*, pages 138–145.
- Markus Freitag and Scott Roy. 2018. Unsupervised natural language generation with denoising autoencoders. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3922–3929.
- Zihao Fu, Lidong Bing, and Wai Lam. 2020a. Open domain event text generation. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 7748–7755.
- Zihao Fu, Lidong Bing, Wai Lam, and Shoaib Jameel. 2020b. Dynamic topic tracker for kb-to-text generation. In *Proceedings of the 28th International Conference on Computational Linguistics: Technical Papers (COLING)*.
- Zihao Fu, Bei Shi, Wai Lam, Lidong Bing, and Zhiyuan Liu. 2020c. Partially-aligned data-to-text generation with distant supervision. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

- Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017a. Creating training corpora for nlg micro-planners. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 179–188.
- Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017b. The webnlg challenge: Generating text from rdf data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133.
- Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. 2016. Dual learning for machine translation. In *Advances in Neural Information Processing Systems*, pages 820–828.
- Vu Cong Duy Hoang, Philipp Koehn, Gholamreza Haffari, and Trevor Cohn. 2018. Iterative back-translation for neural machine translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 18–24.
- Yaser Keneshloo, Tian Shi, Naren Ramakrishnan, and Chandan K Reddy. 2018. Deep reinforcement learning for sequence to sequence models. *arXiv preprint arXiv:1805.09461*.
- Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc’Aurelio Ranzato. 2018a. Unsupervised machine translation using monolingual corpora only. In *International Conference on Learning Representations (ICLR)*.
- Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc’Aurelio Ranzato. 2018b. Phrase-based & neural unsupervised machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Rémi Lebret, David Grangier, and Michael Auli. 2016. Neural text generation from structured data with application to the biography domain. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1203–1213.
- Jiwei Li, Will Monroe, Alan Ritter, Dan Jurafsky, Michel Galley, and Jianfeng Gao. 2016. Deep reinforcement learning for dialogue generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1202.
- Juncen Li, Robin Jia, He He, and Percy Liang. 2018. Delete, retrieve, generate: a simple approach to sentiment and style transfer. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 1865–1874.
- Percy Liang, Michael Jordan, and Dan Klein. 2009. Learning semantic correspondences with less supervision. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 91–99. Association for Computational Linguistics.
- Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. *Workshop on Text Summarization Branches Out*.
- Fuli Luo, Peng Li, Jie Zhou, Pengcheng Yang, Baobao Chang, Zhifang Sui, and Xu Sun. 2019. A dual reinforcement learning framework for unsupervised text style transfer. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence, IJCAI 2019*.
- Christopher Manning, Prabhakar Raghavan, and Hinrich Schütze. 2010. Introduction to information retrieval. *Natural Language Engineering*, 16(1):100–103.
- Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2017. The e2e dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 201–206.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the Annual meeting on Association for Computational Linguistics*, pages 311–318.
- Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96.
- Mingming Sun, Xu Li, and Ping Li. 2018. Logician and orator: Learning from the duality between language and knowledge in open domain. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2119–2130.
- Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112.
- Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 4566–4575.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2017. Challenges in data-to-document generation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2253–2263.

Zili Yi, Hao Zhang, Ping Tan, and Minglun Gong. 2017. Dualgan: Unsupervised dual learning for image-to-image translation. In *Proceedings of the IEEE international conference on computer vision*, pages 2849–2857.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232.

# Modality-Transferable Emotion Embeddings for Low-Resource Multimodal Emotion Recognition

Wenliang Dai, Zihan Liu, Tiezheng Yu, Pascale Fung

Center for Artificial Intelligence Research (CAiRE)

Department of Electronic and Computer Engineering

The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong  
`{wdaiai, zliucr, tyuah}@connect.ust.hk, pascale@ece.ust.hk`

## Abstract

Despite the recent achievements made in the multi-modal emotion recognition task, two problems still exist and have not been well investigated: 1) the relationship between different emotion categories are not utilized, which leads to sub-optimal performance; and 2) current models fail to cope well with low-resource emotions, especially for unseen emotions. In this paper, we propose a modality-transferable model with emotion embeddings to tackle the aforementioned issues. We use pre-trained word embeddings to represent emotion categories for textual data. Then, two mapping functions are learned to transfer these embeddings into visual and acoustic spaces. For each modality, the model calculates the representation distance between the input sequence and target emotions and makes predictions based on the distances. By doing so, our model can directly adapt to the unseen emotions in any modality since we have their pre-trained embeddings and modality mapping functions. Experiments show that our model achieves state-of-the-art performance on most of the emotion categories. Besides, our model also outperforms existing baselines in the zero-shot and few-shot scenarios for unseen emotions<sup>1</sup>.

## 1 Introduction

Multi-modal emotion recognition is an increasingly popular but challenging task. One main challenge is that labelled data is difficult to come by as humans find it time-consuming to discern emotion categories from either speech or video. Indeed we humans express emotions through a combination of modalities, including the way we speak, the words we use, facial expressions and sometimes gestures. It is also much more comfortable for humans to understand each other's emotions when they can both

<sup>1</sup>Code is available at <https://github.com/wenliangdai/Modality-Transferable-MER>

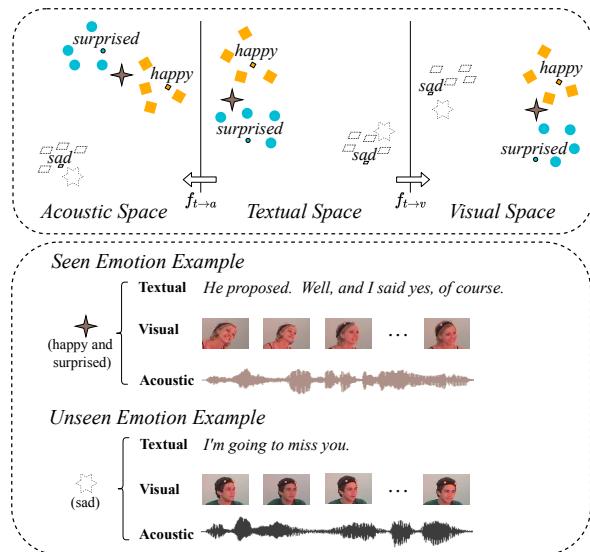


Figure 1: An intuitive example of our method. In the upper image, the relative positions of GloVe emotion embeddings (*happy*, *surprised*) are shown in the textual space, which are then projected to acoustic and visual spaces by two mapping functions ( $f_{t \rightarrow a}$  and  $f_{t \rightarrow v}$ ). Our model learns to group the representations of input sentences (■, ●) based on their corresponding emotion embeddings. Examples are shown in the lower image. When a sample has both *happy* and *surprised* emotions, its representation gets close to these two emotion embeddings in all three spaces. If an unseen emotion *sad* (□) comes, the model processes it with the same flow and recognizes corresponding data samples.

hear and see the other person. It follows that multi-modal emotion recognition can, therefore, yield more reliable results than restricting machines to a single modality.

In the past few years, much research has been done to better understand intra-modality and inter-modality dynamics, and modality fusion is a widely studied approach. For example, [Zadeh et al. \(2017\)](#) proposed a tensor fusion network that combines three modalities from vectors to a tensor using the Cartesian product. In addition, the attention

mechanism is commonly used to do modality fusion (Zadeh et al., 2018a; Wang et al., 2018; Liang et al., 2018; Hazarika et al., 2018; Pham et al., 2018; Tsai et al., 2019a). Although significant improvements have been made on the multi-modal emotion recognition task, however, the relationship between emotions has not been well modelled, which can lead to sub-optimal performance. Also, the problem of low-resource multi-modal emotion recognition is not adequately studied. Multi-modal emotion recognition data is hard to collect and annotate, especially for low-resource emotions (e.g., surprise) that are rarely seen in daily life, which motivates us to investigate this problem.

In this paper, we propose a modality-transferable network with cross-modality emotion embeddings to model the relationship between emotions. Given that emotion embeddings contain semantic information and emotion relations in the vector space, we use them to represent emotion categories and measure the similarity of the representations between the input sentence and target emotions to make predictions. Concretely, for the textual modality, we use the pre-trained GloVe (Pennington et al., 2014) embeddings of emotion words as the emotion embeddings. As there are no pre-trained emotion embeddings for the visual and acoustic modalities, the model learns two mapping functions,  $f_{t \rightarrow v}$  and  $f_{t \rightarrow a}$ , to transfer the emotion embeddings from the textual space to the visual and acoustic spaces (Figure 1). Therefore, for each modality, there will be a dedicated set of emotion embeddings. The distances computed in all modalities will be finally fused, and the model will make predictions based on that.

Benefiting from this prediction mechanism, our model can easily carry out zero-shot learning (ZSL) to identify unseen emotion categories using the embeddings from unseen emotions. The intuition behind it is that the pre-trained and projected emotion embeddings form a semantic knowledge space, which is shared by both the seen and unseen classes. Furthermore, with the help of embedding mapping functions, the model can also perform ZSL on a single modality during inference time. When a few samples from unseen emotions are available, our model can adapt to new emotions without forgetting the previous emotions by using joint training and continual learning (Lopez-Paz and Ranzato, 2017).

Our contributions in this work are three-fold:

- We introduce a simple but effective end-to-end model for the multi-modal emotion recognition task. It learns the relationship of different emotion categories using emotion embeddings.
- To the best of our knowledge, this paper is the first to investigate multi-modal emotion recognition in the low-resource scenario. Our model can directly adapt to an unseen emotion, even if only one modality is available.
- Experimental results show that our model achieves state-of-the-art results on most emotion categories. We also provide a thorough analysis of zero-shot and few-shot learning.

## 2 Related Works

### 2.1 Multi-modal Emotion Recognition

Since the early 2010s, multi-modal emotion recognition has drawn more and more attention with the rise of deep learning and its advances in computer vision and natural language processing (Baltrušaitis et al., 2018). Schuller et al. (2011) proposed the first Audio-Visual Emotion Challenge and Workshop (AVEC), which focused on multi-modal emotion analysis for health. In recent years, most achievements in this area aimed to find a better modality fusion method. Zadeh et al. (2017) introduced a tensor fusion network that combined data representation from each modality to a tensor by performing the Cartesian product. In addition, the attention mechanism (Bahdanau et al., 2015) has been widely applied to do modality fusion and emphasis (Zadeh et al., 2018a; Pham et al., 2018; Tsai et al., 2019a). Furthermore, Liu et al. (2018) proposed a low-rank architecture to decrease the problem complexity, and Tsai et al. (2019b) introduced a modality re-construction method to generate occasional missing data in a modality.

Although prior works have made progress on this task, the relationship between emotion categories has not been well modelled in previous works, except by Xu et al. (2020), who captured emotion correlations using graph networks for emotion recognition. However, the model is only based on a single textual modality. Additionally, the previous studies have not put much effort toward unseen and low-resource emotion categories, which is a problem of multi-modal emotion data by nature.

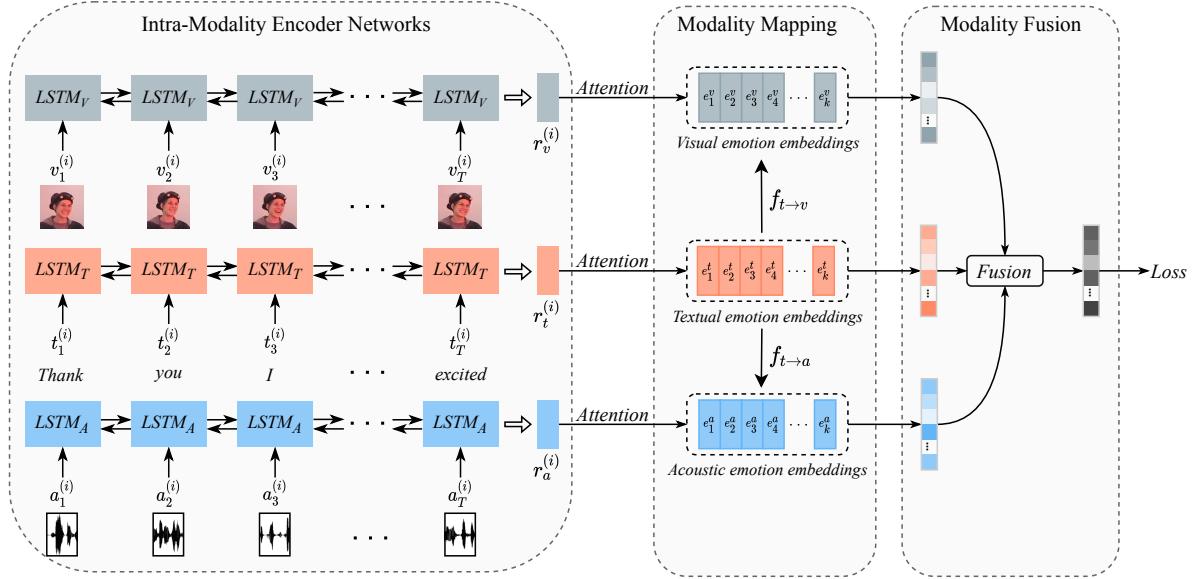


Figure 2: The architecture of our proposed multi-modal emotion recognition model. It consists of three LSTM networks, one emotion embedding mapping module, and one modality fusion module. For each modality, the input is a sequence of length  $T$ . Each modality has a set of emotion embeddings by mapping the GloVe textual emotion embeddings to the other modalities using  $f_{t \rightarrow v}$  and  $f_{t \rightarrow a}$ . The whole architecture is optimized end-to-end.

## 2.2 Zero/Few-Shot and Continual Learning

Zero-shot and few-shot learning methods, which address the data scarcity scenario, have been applied to many popular machine learning tasks where zero or only a few training samples are available for the target tasks or domains, such as machine translation (Johnson et al., 2017; Gu et al., 2018), dialogue generation (Zhao and Eskenazi, 2018; Madotto et al., 2019), dialogue state tracking (Liu et al., 2019c; Wu et al., 2019), slot filling (Bapna et al., 2017; Liu et al., 2019b, 2020), and accented speech recognition (Winata et al., 2020). They have also been adopted in multiple cross-lingual tasks, such as named entity recognition (Xie et al., 2018; Ni et al., 2017), part-of-speech tagging (Wisniewski et al., 2014; Huck et al., 2019), and question answering (Liu et al., 2019a; Lewis et al., 2019). Recently, several methods have been proposed for continual learning (Rusu et al., 2016; Kirkpatrick et al., 2017; Lopez-Paz and Ranzato, 2017; Fernando et al., 2017; Lee et al., 2017), and these were applied to some NLP tasks, such as opinion mining (Shu et al., 2016), document classification (Shu et al., 2017), and dialogue state tracking (Wu et al., 2019).

## 3 Methodology

As shown in Figure 2, our model consists of three parts: intra-modality encoder networks, emotion em-

bedding mapping modules, and an inter-modal fusion module. In this section, we first define the problem, and then we introduce the details of our model.

### 3.1 Problem Definition

We define the input multi-modal data samples as  $X = \{(t_i, a_i, v_i)\}_{i=1}^I$ , in which  $I$  denotes the total number of samples, and  $t, a, v$  denote the *textual*, *acoustic*, and *visual* modalities, respectively. For each modality, there is a set of emotion embeddings that represent the semantic meanings for the emotion categories to be recognized. In the textual modality, we have  $E_t = \{e_k^t\}_{k=1}^K$ , which is from the pre-trained GloVe embeddings. In acoustic and visual modalities, we have  $E_a = \{e_k^a\}_{k=1}^K$  and  $E_v = \{e_k^v\}_{k=1}^K$ , which are mapped from  $E_t$  by the mapping function  $f_{t \rightarrow v}$  and  $f_{t \rightarrow a}$ .  $K$  denotes the number of emotion categories, and it can be changed to fit different tasks and zero-shot learning.  $Y = \{y_i\}_{i=1}^I$  denotes the annotations for multi-label emotion recognition, where  $y_i$  is a vector of length  $K$  with binary values.

### 3.2 Intra-modality Encoder Networks

As shown in Figure 2, for each data sample, there are three sequences of length  $T$  from the three modalities. For each modality, we use a bi-directional Long-Short Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) net-

work as the encoder to process the sequence and get a vector representation. In other words, for the  $i^{th}$  data sample, we will have three vectors,  $r_t^{(i)} \in \mathbb{R}^{d_t}$ ,  $r_a^{(i)} \in \mathbb{R}^{d_a}$ , and  $r_v^{(i)} \in \mathbb{R}^{d_v}$ , that represent the *textual*, *acoustic*, and *visual* modalities. Here,  $d_t$ ,  $d_a$ , and  $d_v$  are the dimensions of the *textual*, *acoustic*, and *visual* modalities, respectively.

### 3.3 Modality Mapping Module

As mentioned in Section 1, previous works do not consider the connections in different emotion categories, and the only information about emotions is in the annotations. In our model, we use emotion word embeddings to inject the semantic information of emotions into the model. Additionally, emotion embeddings also contain the relationships between emotion categories. For the textual modality, we use pre-trained GloVe (Pennington et al., 2014) embeddings of  $K$  emotion words, denoted as  $E_t \in \mathbb{R}^{K \times d_t}$ . For the other two modalities, because there are no off-the-shelf pre-trained emotion embeddings, our model learns two mapping functions which project the vectors from the textual space into the acoustic and visual spaces:

$$E_a = f_{t \rightarrow a}(E_t) \in \mathbb{R}^{K \times d_a} \quad (1)$$

$$E_v = f_{t \rightarrow v}(E_t) \in \mathbb{R}^{K \times d_v}. \quad (2)$$

### 3.4 Modality Fusion and Prediction

To predict the emotions for input sentences, we calculate the similarity scores between the sequence representation and the emotion embeddings for each modality. As shown in Eq.3, for a data sample  $i$ , every modality will have a vector of similarity scores of length  $K$  by dot product attention. We further add a modality fusion module to weighted sum all the vectors, in which the weights are also optimized end-to-end (Eq.4). Finally, as the datasets are multi-labelled, the sigmoid activation function is applied to each score in the fused vector  $s^{(i)}$ , and a threshold is used to decide whether an emotion exists or not.

$$s_t^{(i)} = E_t r_t^{(i)}, s_a^{(i)} = E_a r_a^{(i)}, s_v^{(i)} = E_v r_v^{(i)} \quad (3)$$

$$s^{(i)} = \text{Sigmoid}(w_t s_t^{(i)} + w_a s_a^{(i)} + w_v s_v^{(i)}) \quad (4)$$

## 4 Unseen Emotion Prediction

Collecting numerous training samples for a new emotion, especially for a low-resource emotion, is expensive and time-consuming. Therefore, in

this section, we concentrate on the ability of our model to generalize to an unseen target emotion by considering the scenario where we have zero or only a few training samples in an unseen emotion.

### 4.1 Zero-Shot Emotion Prediction

Ideally, our model is able to directly adapt to a new emotion based on its embedding. Given a new text emotion embedding  $e_{k+1}^t$ , we can generate the visual and acoustic emotion embeddings  $e_{k+1}^v$  and  $e_{k+1}^a$ , respectively, using the already learned mapping functions  $f_{t \rightarrow v}$  and  $f_{t \rightarrow a}$ . After that, the similarity scores between the input sentence and the new emotion can be computed for each modality.

### 4.2 Few-Shot Emotion Prediction

In this section, we assume 1% of the positive training samples in a new emotion are available, and to balance the training samples, we take the same amount of negative training samples for the new emotion. However, the model could lose its ability to predict the original emotions when we simply fine-tune it on the training samples of a new emotion. To cope with this issue, we propose two fine-tuning settings. First, after we obtain the trained model in the source emotions, we jointly train it with the training samples of the source emotions and the new target emotion. Second, we utilize a continual learning method, gradient episodic memory (GEM) (Lopez-Paz and Ranzato, 2017), to prevent the catastrophic forgetting of previously learned knowledge. The purpose of using continual learning is that we do not need to retrain with all the data from previously learned emotions since the data might not be available. We describe the training process of GEM as follows:

We define  $\Theta_S$  as the model’s parameters trained in the source emotions, and  $\Theta$  denotes the current optimized parameters based on the target emotion data. GEM keeps a small number of samples  $N$  from the source emotions, and a constraint is applied on the gradient to prevent the loss on the stored samples from increasing when the model learns the new target emotion. The training process can be formulated as

$$\text{Minimize}_{\Theta} L(\Theta)$$

$$\text{Subject to } L(\Theta, N) \leq L(\Theta_S, N),$$

where  $L(\Theta, N)$  is the loss value of the  $N$  stored samples.

## 5 Experiments

In this section, we first introduce the two public datasets we use and data feature extraction. Then, we discuss our evaluation metrics, including their advantages and defects. Finally, we introduce the baselines and our experimental settings.

### 5.1 Datasets

**CMU-MOSEI** CMU Multimodal Opinion Sentiment and Emotion Intensity (CMU-MOSEI) (Zadeh et al., 2018b) is currently the largest public dataset for multi-modal sentiment analysis and emotion recognition. It comprises 23,453 annotated data samples extracted from 3228 videos. For emotion recognition, it consists of six basic categories: *anger*, *disgust*, *fear*, *happy*, *sad*, and *surprise*. For zero-shot and few-shot learning evaluation, we use four relatively low-resource categories among them (*anger*, *disgust*, *fear*, *surprise*). The model is trained on the other five categories when evaluating one zero-shot category. A detailed statistical table about these categories is included in Appendix A.

**IEMOCAP** The Interactive Emotional Dyadic Motion Capture (IEMOCAP) (Busso et al., 2008) dataset was created for multi-modal human emotion analysis, and was collected from dialogues performed by ten actors. It is also a multi-labelled emotion recognition dataset which contains nine emotion categories. For comparison with prior works (Wang et al., 2018; Liang et al., 2018; Pham et al., 2018; Tsai et al., 2019a) where four (out of the nine) emotion categories are selected for training and evaluating the models, we also follow the same four categories, namely, *happy*, *sad*, *angry*, and *neutral*, to train our model. For zero-shot learning evaluation, we consider three low-resource categories from the remaining five, namely, *excited*, *surprised*, and *frustrated*, as unseen emotions.

### 5.2 Data Feature Extraction

We use CMU-Multimodal SDK (Zadeh et al., 2018c) for downloading and pre-processing the datasets. It helps to do data alignment and early-stage feature extraction for each modality. The textual data is tokenized in word level and represented using GloVe (Pennington et al., 2014) embeddings. Facial action units are extracted by the Facet (iMotions, 2017) to indicate muscle movements and expressions (Ekman et al., 1980). These are a commonly used type of feature for facial expression

recognition (Fan et al., 2020). For acoustic data, COVAREP (Degottex et al., 2014) is used to extract fundamental features, such as mel-frequency cepstral coefficients (MFCCs), pitch tracking, glottal source parameters, etc.

### 5.3 Evaluation Metrics

**Weighted Accuracy** Due to the imbalanced nature of the emotion recognition dataset (for each emotion category, there are many more negative samples than positive samples), we use binary weighted accuracy (Tong et al., 2017) on each category to better measure the model’s performance. The formula is

$$\text{Weighted Acc.} = \frac{TP \times N/P + TN}{2N}$$

where P means total positive, TP true positive, N total negative, and TN true negative.

**Weighted F1** In prior works (Zadeh et al., 2018b; Akhtar et al., 2019; Tsai et al., 2019a), the binary weighted F1 score metric is used on the CMU-MOSEI dataset, and its formula is shown in Eq.5.

$$\text{Weighted F1} = \frac{P}{I} \times F1_p + \frac{N}{I} \times F1_n \quad (5)$$

Here,  $F1_p$  is the F1 score that treats positive samples as *positive*, while  $F1_n$  treats negative samples as *positive*, and they are weighted by their portion of the data. However, there is one defect of using binary weighted F1 in this task. As there are many more negative samples than positive ones, we find that with the increase of the threshold, the weighted F1 score will also increase because the *true negative* increases. Therefore, in this paper, we do not report this metric. A detailed analysis of this is given in Appendix B.

**AUC Score** To eliminate the effect of threshold and mitigate the defect of the weighted F1 score, we also report Area under the ROC Curve (AUC) scores. The AUC score considers classification performance on both positive and negative samples, and it is scale- and threshold-invariant.

### 5.4 Baselines

For both the CMU-MOSEI and IEMOCAP datasets, we use Early Fusion LSTM (EF-LSTM) and Late Fusion LSTM (LF-LSTM) as two baseline models. Additionally, for CMU-MOSEI, the Graph Memory Fusion Network (Graph-MFN) (Zadeh et al., 2018b) and a multi-task learning (MTL)

Emotion	Anger		Disgust		Fear		Happy		Sad		Surprise		Average	
Metrics	W-Acc	AUC												
EF-LSTM	58.5	62.2	59.9	63.9	50.1	69.8	65.1	68.9	55.1	58.6	50.6	54.3	56.6	63.0
LF-LSTM	57.7	66.5	61.0	71.9	50.7	61.1	63.9	68.6	54.3	59.6	51.4	61.5	56.5	64.9
Graph-MFN	62.6	-	69.1	-	62.0	-	66.3	-	60.4	-	53.7	-	62.3	-
MTL	66.8	68.0 <sup>†</sup>	<b>72.7</b>	76.7 <sup>†</sup>	62.2	42.9 <sup>†</sup>	53.6	71.4 <sup>†</sup>	61.4	57.6 <sup>†</sup>	60.6	65.1 <sup>†</sup>	62.8	63.6 <sup>†</sup>
Ours	<b>67.0</b>	<b>71.7</b>	72.5	<b>78.3</b>	<b>65.4</b>	<b>71.6</b>	<b>67.9</b>	<b>73.9</b>	<b>62.6</b>	<b>66.7</b>	<b>62.1</b>	<b>66.4</b>	<b>66.2</b>	<b>71.4</b>

Table 1: Results of multi-modal emotion recognition on the CMU-MOSEI dataset. Baselines (EF-LSTM, LF-LSTM) and previous state-of-the-art models (Graph-MFN (Zadeh et al., 2018b), MTL (Akhtar et al., 2019)) are compared. Results marked by <sup>†</sup> are re-run and fine-tuned by us as they are not reported in the original paper.

Emotion	Happy		Sad		Angry		Neutral	
	Acc	AUC	Acc	AUC	Acc	AUC	Acc	AUC
EF-LSTM	85.8	70.7	83.7	85.8	75.8	90.3	67.1	74.1
LF-LSTM	85.2	71.7	83.4	84.4	79.5	86.8	66.5	72.2
RMFN (Liang et al., 2018)	<b>87.5</b>	-	83.8	-	85.1	-	69.5	-
RAVEN (Wang et al., 2018)	87.3	-	83.4	-	87.3	-	69.7	-
MCTN (Pham et al., 2018)	84.9	-	80.5	-	79.7	-	62.3	-
MulT (Tsai et al., 2019a)	83.5 <sup>†</sup>	71.2 <sup>†</sup>	85.0 <sup>†</sup>	<b>89.3<sup>†</sup></b>	85.5 <sup>†</sup>	92.4 <sup>†</sup>	71.0 <sup>†</sup>	<b>77.2<sup>†</sup></b>
Ours	85.0	<b>74.2</b>	<b>86.6</b>	88.4	<b>88.1</b>	<b>93.2</b>	<b>71.1</b>	76.7

Table 2: Multi-modal emotion recognition results on IEMOCAP. We re-run MulT (marked by <sup>†</sup>) with its reported best hyper-parameters to get the AUC scores.

model (Akhtar et al., 2019) are included for comparison with previous state-of-the-art models. For IEMOCAP, the Recurrent Multistage Fusion Network (RMFN) (Liang et al., 2018), Recurrent Attended Variation Embedding Network (RAVEN) (Wang et al., 2018), and the Multimodal Transformer (MulT) (Tsai et al., 2019a) are included. To compare the AUC scores and zero-shot performance with baselines, we re-run the MTL and MulT models based on their reported best hyper-parameters, and we also carry out hyper-parameter search for a fair comparison.

	CMU-MOSEI	IEMOCAP
Best Epoch	15	16
Batch size	512	32
Learning rate	1e-4	1e-3
# LSTM layers	2	2
Hidden Size	300/200/100	300/200/100
Dropout	0.15	0.15
Gradient Clip	10.0	1.0
Random Seed	0	0

Table 3: The hyper-parameters of our best models. The hidden size means the size of the LSTM hidden state of the textual/acoustic/visual modality, respectively.

## 5.5 Training Details

The model is trained end-to-end with the Adam optimizer (Kingma and Ba, 2015) and a scheduler that will reduce the learning rate by a factor of 0.1 when the optimization stays on a plateau for more than 5 epochs. The best hyper-parameters in

our training for both datasets are shown in Table 3. Also, we use the largest GloVe word embeddings (glove.840B.300d<sup>2</sup>) for both the input text data and the emotion embeddings in the textual modality. The weights of the textual embeddings are frozen during training to keep the pre-trained relations, which is also essential for doing zero-shot learning.

## 6 Analysis

### 6.1 Results

Table 1 shows our model’s performance on the CMU-MOSEI dataset. Compared to existing baselines, our model surpasses them by a large margin. The weighted accuracy (W-Acc) and AUC score are used for evaluation, with a threshold set to 0.5 to calculate the W-Acc. As discussed in Section 5.3, we do not follow the previous papers in using the weighted F1-score (W-F1) because it does not provide an effective evaluation when the dataset is very imbalanced. For example, the weakest baseline, EF-LSTM, can even achieve 90% W-F1 by predicting almost all samples as negative. More plots and analysis of this defect of W-F1 are included in Appendix B.

We further test our model on a second dataset called IEMOCAP, and the results are shown in Table 2. Similarly, our model achieves better results on most emotion categories, except *happy*. For a

<sup>2</sup><https://nlp.stanford.edu/projects/glove/>

Metrics		W-Acc	AUC	W-Acc	AUC	W-Acc	AUC	W-Acc	AUC
Unseen emotion		Anger (unseen)		Disgust (unseen)		Fear (unseen)		Surprise (unseen)	
Zero-Shot	EF-LSTM	50.6	50.9	50.3	48.2	45.8	42.3	50.2	46.9
	LF-LSTM	48.4	49.2	49.7	44.2	<b>47.4</b>	<b>47.3</b>	48.6	48.3
	Ours	55.9	61.6	67.5	72.7	41.8	40.6	53.4	55.5
1% Few-Shot	FT (Ours)	58.9	<b>61.9</b>	67.9	71.5	43.1	43.1	51.8	53.9
	CL (Ours)	58.9	61.5	68.7	72.8	42.6	42.7	50.6	52.5
	JT (Ours)	<b>59.0</b>	61.1	<b>69.2</b>	<b>74.2</b>	41.9	41.7	<b>55.2</b>	<b>58.1</b>
Average on all categories		Except Anger		Except Disgust		Except Fear		Except Surprise	
Zero-shot	Ours	65.6	70.6	64.4	69.3	65.9	70.9	67.2	71.4
1% Few-Shot	FT (Ours)	64.4	<b>69.8</b>	63.7	68.5	65.4	70.7	65.1	71.4
	CL (Ours)	<b>64.6</b>	<b>69.8</b>	<b>63.8</b>	<b>68.9</b>	65.6	<b>70.9</b>	65.5	<b>71.5</b>
	JT (Ours)	64.3	69.3	63.5	68.8	<b>65.9</b>	70.8	<b>66.1</b>	<b>71.5</b>

Table 4: Zero/few-shot results on low-resource emotion categories in CMU-MOSEI dataset. Here, FT, CL, and JT stand for *Fine-Tuning*, *Continual Learning*, and *Joint Training* respectively. FT directly fine-tunes the trained model on the unseen emotions, and CL and JT are two different settings introduced in Section 4.2. Note that in the few-shot settings, we select the model based on the average performance of all emotions (including the unseen emotion) to ensure good overall performance of our model.

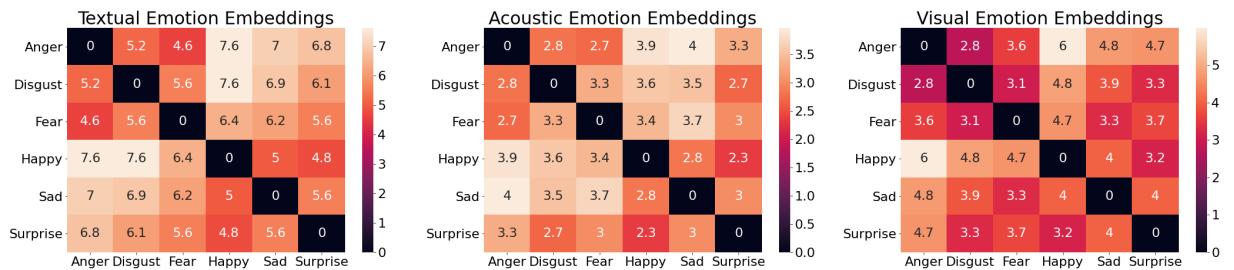


Figure 3: Euclidean distances between different emotion embeddings in the textual, acoustic, and visual spaces. Although the absolute values are different, the relative distances between emotion categories are well reserved. This indicates that the two mapping functions  $f_{t \rightarrow v}$  and  $f_{t \rightarrow a}$  transfer the relationships of emotion categories well.

fair comparison on IEMOCAP, we use accuracy instead of W-Acc, following the previous works compared in the table.

## 6.2 Effects of Emotion Embeddings

Quantitatively, our model makes a large improvement in the multi-modal emotion recognition task. We think it benefits greatly from the emotion embeddings, which can model the relationships (or distances) between emotion categories. This is especially important for emotion recognition, which is a multi-label task by nature, as people can have multiple emotions at the same time. For example, if a person is *surprised*, it is more likely that this person is also *happy* and *excited* and is less likely to be *disgusted* or *sad*. This kind of information is expected to be modelled and captured by emotion embeddings. Intuitively, in the textual space, related emotions (e.g., *angry* and *disgusted*) tend to have closer word vectors than unrelated emotions (*angry* and *happy*). To ensure the effectiveness of word embeddings, for each emotion word, we investi-

gated multiple forms of it. For example, for *surprised*, we also tried with *Surprised*, *(S/s)urprising*, *(S/s)urprise*. Generally, they all show a similar trend, and in most cases, the word form that is used to describe human shows the best results. In our final setting, we iterate and pick the best performing form for each emotion category.

Moreover, our model can transfer the relationship of emotion categories from the textual space to the acoustic and visual spaces using end-to-end optimized mapping functions. In Figure 3, we show the Euclidean distances of emotion embeddings between categories. The relative positions are preserved very well after being transferred from the textual space to the visual and acoustic spaces. This indicates that the learned mapping functions ( $f_{t \rightarrow v}$  and  $f_{t \rightarrow a}$ ) are effective. Although it is not the main focus of this paper, we think improving the pre-trained textual emotion embeddings is an essential direction for future work. It can benefit all modalities and further enhance the overall performance. For example, incorporate semantic emotion

information (Xu et al., 2018) to the original word embeddings.

### 6.3 Zero/Few-Shot Results

Benefiting from the pre-trained textual emotion embeddings and learned mapping functions, our model can recognize unseen emotion categories to a certain extent. We evaluate our model’s zero-shot learning ability on the low-resource categories in CMU-MOSEI (shown in Table 4) and IEMOCAP (shown in Table 5). For a fair comparison, we use the same training setting that is used in Table 1. This can ensure that no downgrade happens on the seen emotions, and the model is not selected to overfit a single unseen category. As we can see, the zero-shot results of the baselines are similar to random guesses, because the weights related to that unseen emotion in the model are randomly initialized and have never been optimized. For our model, the zero-shot performance is much better than that of the baselines in almost all emotions. This is because our model learns to classify emotion categories based on the similarity between the sentence representation and emotion embeddings, which enables our model better generalization ability to other unseen emotions since emotion embeddings contain semantic information in the vector space.

Furthermore, we perform few-shot learning using only 1% of data of these low-resource categories. As we can see from Table 4, using very few training samples, our model can adapt to unseen emotions without losing the performance in the source emotions. In addition, we observe that simply fine-tuning (FT) our trained model sometimes obtains inferior performance. This is because our model will gradually lose the ability to classify the source emotions, and we have to early stop the fine-tuning process, which leads to inferior performance. We can see that CL and JT prevent our model from catastrophic forgetting and improve the few-shot performance in the unseen emotion. Moreover, JT achieves slightly better performance than CL. This can be attributed to the fact that CL might still result in performance drops in source emotions since our model only observes partial samples from them. At the same time, JT directly optimizes the model on the data samples of such emotions.

Unseen emotion	Excited (unseen)		Surprised (unseen)		Frustrated (unseen)	
Metrics	Acc	F1	Acc	F1	Acc	F1
EF-LSTM	13.1	23.1	11.3	5.1	22.9	37.2
LF-LSTM	14.0	23.3	2.6	5.1	23.7	37.4
MuLT	45.1	27.3	41.4	7.5	48.7	40.9
Ours (TAV)	82.0	56.1	78.8	13.1	73.6	57.9
Ours (TA)	79.9	52.7	79.3	14.4	75.1	60.1
Ours (TV)	75.9	42.7	58.6	9.1	54.1	13.6
Ours (AV)	89.1	69.9	65.7	13.2	83.9	73.6
Ours (T)	72.9	37.1	67.7	3.1	55.3	9.0
Ours (A)	76.9	52.8	82.1	16.8	86.1	74.8
Ours (V)	82.1	35.0	81.1	6.2	68.6	44.6

Table 5: Zero-shot results on the IEMOCAP dataset. T (textual), A (acoustic), and V (visual) indicate the existence of that modality during inference time.

Metric	W-Acc					
	Anger	Disgust	Fear	Happy	Sad	Surprise
T+A+V	<b>67.0</b>	<b>72.5</b>	65.4	<b>67.9</b>	62.6	<b>62.1</b>
T+A	65.0	71.9	64.8	66.0	<b>63.0</b>	59.9
T+V	64.9	71.2	<b>66.7</b>	67.6	61.0	60.4
A+V	63.8	71.1	65.5	64.5	61.3	55.2
Only T	61.5	69.0	64.3	64.2	59.7	61.2
Only A	61.9	71.5	66.9	62.7	61.0	54.8
Only V	63.4	69.7	63.2	63.2	58.5	53.3

Table 6: Ablation study on CMU-MOSEI dataset. Different combinations of subsets of modalities are used.

### 6.4 Ablation Study

To further investigate how each individual modality influences the model, we perform comprehensive ablation studies on supervised multi-modal emotion recognition and also zero-shot prediction.

In Table 6, we enumerate different subsets of the (*textual*, *acoustic*, *visual*) modalities to evaluate the effect of each single modality. Generally, the performance will increase if more modalities are available. Compared to single-modal data, multi-modal data can provide supplementary information, which leads to more accurate emotion recognition. In terms of a single modality, we find that *textual* and *acoustic* are more effective than *visual*.

Similarly, in Table 5, we show the zero-shot performance with different combinations of modalities during the inference time (all modalities exist in the training phase). As there are many more negative samples than positive ones in the ZSL setting, we also evaluate the models with the unweighted F1 score. Because if a model has high accuracy but a low F1, it is heavily biased to the negative samples so it cannot do classification effectively. Empirical results indicate that zero-shot on only one modality is possible. Moreover, if the data of an emotion category has strong characteristics in

one modality and is ambiguous in other modalities, single-modality can even surpass multi-modality on zero-shot prediction. For example, the performance of single-modality zero-shot prediction using the *acoustic* modality on the *surprised* category is better than using all modalities.

## 7 Conclusion

In this paper, we introduce a modality-transferable model that leverages cross-modality emotion embeddings for multi-modal emotion recognition. It makes predictions by measuring the distances between input data and target emotion categories, which is especially effective for a multi-label problem. The model also learns two mapping functions to transfer pre-trained textual emotion embeddings to acoustic and visual spaces. The empirical results demonstrate that it exhibits state-of-the-art performance on most of the categories. Enabled by the utilization of emotion embeddings, our model can carry out zero-shot learning for unseen emotion categories and can quickly adapt few-shot learning without downgrading trained categories.

## Acknowledgement

This work is funded by MRP/055/18 of the Innovation Technology Commission, the Hong Kong SAR Government.

## References

- Md Shad Akhtar, Dushyant Chauhan, Deepanway Ghosal, Soujanya Poria, Asif Ekbal, and Pushpak Bhattacharyya. 2019. Multi-task learning for multi-modal emotion recognition and sentiment analysis. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 370–379, Minneapolis, Minnesota. Association for Computational Linguistics.
- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.
- Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. 2018. Multimodal machine learning: A survey and taxonomy. *IEEE transactions on pattern analysis and machine intelligence*, 41(2):423–443.
- Ankur Bapna, Gokhan Tür, Dilek Hakkani-Tür, and Larry Heck. 2017. Towards zero-shot frame semantic parsing for domain scaling. *Proc. Interspeech 2017*, pages 2476–2480.
- Carlos Busso, Murtaza Bulut, Chi-Chun Lee, Abe Kazemzadeh, Emily Mower Provost, Samuel Kim, Jeannette N. Chang, Sungbok Lee, and Shrikanth S. Narayanan. 2008. Iemocap: interactive emotional dyadic motion capture database. *Language Resources and Evaluation*, 42:335–359.
- Gilles Degottex, John Kane, Thomas Drugman, Tuomo Raitio, and Stefan Scherer. 2014. Covarep — a collaborative voice analysis repository for speech technologies. *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 960–964.
- Paul Ekman, Wallace V Freisen, and Sonia Ancoli. 1980. Facial signs of emotional experience. *Journal of personality and social psychology*, 39(6):1125.
- Yingruo Fan, Jacqueline Lam, and Victor Li. 2020. Facial action unit intensity estimation via semantic correspondence learning with dynamic graph convolution. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*.
- Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A Rusu, Alexander Pritzel, and Daan Wierstra. 2017. Pathnet: Evolution channels gradient descent in super neural networks. *arXiv preprint arXiv:1701.08734*.
- Jiatao Gu, Yong Wang, Yun Chen, Victor OK Li, and Kyunghyun Cho. 2018. Meta-learning for low-resource neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3622–3631.
- Devamanyu Hazarika, Soujanya Poria, Amir Zadeh, Erik Cambria, Louis-Philippe Morency, and Roger Zimmermann. 2018. Conversational memory network for emotion recognition in dyadic dialogue videos. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2122–2132, New Orleans, Louisiana. Association for Computational Linguistics.
- Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9:1735–1780.
- Matthias Huck, Diana Dutka, and Alexander Fraser. 2019. Cross-lingual annotation projection is effective for neural part-of-speech tagging. In *Proceedings of the Sixth Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 223–233.
- iMotions. 2017. Facial expression analysis. <https://imotions.com/biosensor/fea-facial-expression-analysis/>. Accessed: 2010-09-30.
- Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al. 2017. Google’s multilingual neural machine

- translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.
- Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.
- James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526.
- Sang-Woo Lee, Jin-Hwa Kim, Jaehyun Jun, Jung-Woo Ha, and Byoung-Tak Zhang. 2017. Overcoming catastrophic forgetting by incremental moment matching. In *Advances in neural information processing systems*, pages 4652–4662.
- Patrick Lewis, Barlas Oğuz, Ruty Rinott, Sebastian Riedel, and Holger Schwenk. 2019. Mlqa: Evaluating cross-lingual extractive question answering. *arXiv preprint arXiv:1910.07475*.
- Paul Pu Liang, Ziyin Liu, AmirAli Bagher Zadeh, and Louis-Philippe Morency. 2018. Multimodal language analysis with recurrent multistage fusion. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 150–161, Brussels, Belgium. Association for Computational Linguistics.
- Jiahua Liu, Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2019a. Xqa: A cross-lingual open-domain question answering dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2358–2368.
- Zhun Liu, Ying Shen, Varun Bharadhwaj Lakshminarasimhan, Paul Pu Liang, AmirAli Bagher Zadeh, and Louis-Philippe Morency. 2018. Efficient low-rank multimodal fusion with modality-specific factors. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2247–2256, Melbourne, Australia. Association for Computational Linguistics.
- Zihan Liu, Jamin Shin, Yan Xu, Genta Indra Winata, Peng Xu, Andrea Madotto, and Pascale Fung. 2019b. Zero-shot cross-lingual dialogue systems with transferable latent variables. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1297–1303.
- Zihan Liu, Genta Indra Winata, Zhaojiang Lin, Peng Xu, and Pascale Fung. 2019c. Attention-informed mixed-language training for zero-shot cross-lingual task-oriented dialogue systems. *arXiv preprint arXiv:1911.09273*.
- Zihan Liu, Genta Indra Winata, Peng Xu, and Pascale Fung. 2020. Coach: A coarse-to-fine approach for cross-domain slot filling. *arXiv preprint arXiv:2004.11727*.
- David Lopez-Paz and Marc’Aurelio Ranzato. 2017. Gradient episodic memory for continual learning. In *Advances in neural information processing systems*, pages 6467–6476.
- Andrea Madotto, Zhaojiang Lin, Chien-Sheng Wu, and Pascale Fung. 2019. Personalizing dialogue agents via meta-learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5454–5459.
- Jian Ni, Georgiana Dinu, and Radu Florian. 2017. Weakly supervised cross-lingual named entity recognition via effective annotation and representation projection. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1470–1480.
- Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.
- Hai Pham, Paul Pu Liang, Thomas Manzini, Louis-Philippe Morency, and Barnabás Póczos. 2018. Found in translation: Learning robust joint representations by cyclic translations between modalities. *ArXiv*, abs/1812.07809.
- Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. 2016. Progressive neural networks. *arXiv preprint arXiv:1606.04671*.
- Björn Schuller, Michel Valstar, Florian Eyben, Gary McKeown, Roddy Cowie, and Maja Pantic. 2011. Avec 2011—the first international audio/visual emotion challenge. In *Affective Computing and Intelligent Interaction*, pages 415–424, Berlin, Heidelberg. Springer Berlin Heidelberg.
- Lei Shu, Bing Liu, Hu Xu, and Annice Kim. 2016. Lifelong-rl: Lifelong relaxation labeling for separating entities and aspects in opinion targets. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, Conference on Empirical Methods in Natural Language Processing*, volume 2016, page 225. NIH Public Access.
- Lei Shu, Hu Xu, and Bing Liu. 2017. Doc: Deep open classification of text documents. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2911–2916.
- Edmund Tong, Amir Zadeh, Cara Jones, and Louis-Philippe Morency. 2017. Combating human trafficking with multimodal deep models. In *Proceedings*

*of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1547–1556, Vancouver, Canada. Association for Computational Linguistics.

Yao-Hung Hubert Tsai, Shaojie Bai, Paul Pu Liang, J. Zico Kolter, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2019a. Multimodal transformer for unaligned multimodal language sequences. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Florence, Italy. Association for Computational Linguistics.

Yao-Hung Hubert Tsai, Paul Pu Liang, Amir Zadeh, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2019b. Learning factorized multimodal representations. *ArXiv*, abs/1806.06176.

Yansen Wang, Ying Shen, Zhun Liu, Paul Pu Liang, Amir Zadeh, and Louis-Philippe Morency. 2018. Words can shift: Dynamically adjusting word representations using nonverbal behaviors. *Proceedings of the ... AAAI Conference on Artificial Intelligence. AAAI Conference on Artificial Intelligence*, 33 1:7216–7223.

Genta Indra Winata, Samuel Cahyawijaya, Zihan Liu, Zhaojiang Lin, Andrea Madotto, Peng Xu, and Pascale Fung. 2020. Learning fast adaptation on cross-accented speech recognition. *arXiv preprint arXiv:2003.01901*.

Guillaume Wisniewski, Nicolas Pécheux, Souhir Gahbiche-Braham, and François Yvon. 2014. Cross-lingual part-of-speech tagging through ambiguous learning. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1779–1785.

Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, and Pascale Fung. 2019. Transferable multi-domain state generator for task-oriented dialogue systems. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 808–819.

Jiateng Xie, Zhilin Yang, Graham Neubig, Noah A Smith, and Jaime G Carbonell. 2018. Neural cross-lingual named entity recognition with minimal resources. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 369–379.

Peng Xu, Zihan Liu, Genta Indra Winata, Zhaojiang Lin, and Pascale Fung. 2020. Emograph: Capturing emotion correlations using graph networks. *arXiv preprint arXiv:2008.09378*.

Peng Xu, Andrea Madotto, Chien-Sheng Wu, Ji Ho Park, and Pascale Fung. 2018. Emo2vec: Learning generalized emotion representation by multi-task training. *arXiv preprint arXiv:1809.04505*.

Amir Zadeh, Minghai Chen, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. 2017. Tensor fusion network for multimodal sentiment analysis. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1103–1114, Copenhagen, Denmark. Association for Computational Linguistics.

Amir Zadeh, Paul Pu Liang, Navonil Mazumder, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. 2018a. Memory fusion network for multi-view sequential learning. In *AAAI*.

Amir Zadeh, Paul Pu Liang, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. 2018b. Multimodal language analysis in the wild: Cmu-mosei dataset and interpretable dynamic fusion graph. In *ACL*.

Amir Zadeh, Paul Pu Liang, Soujanya Poria, Prateek Vij, Erik Cambria, and Louis-Philippe Morency. 2018c. Multi-attention recurrent network for human communication comprehension. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Tiancheng Zhao and Maxine Eskenazi. 2018. Zero-shot dialog generation with cross-domain latent actions. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 1–10.

## A Statistics of Datasets

	Train	Valid	Test
Anger	3443	427	971
Disgust	2720	352	922
Fear	1319	186	332
Happy	8147	1313	2522
Sad	3906	576	1334
Surprise	1562	201	479

Table 7: Statistics of the CMU-MOSEI dataset. Some emotion categories are very low-resource.

	Train	Valid	Test
Happy	338	116	135
Sad	690	188	193
Angry	735	136	227
Neutral	954	358	383
Excited	-	-	141
Surprised	-	-	25
Frustrated	-	-	278

Table 8: Statistics of emotion categories in the IEMOCAP dataset. The three at the bottom are unseen emotions used for the evaluation of zero-shot learning.

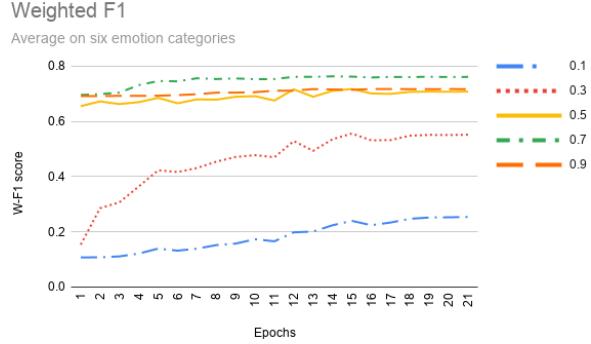


Figure 4: Trend lines of the weighted f1 (W-F1) score during training on the validation set of CMU-MOSEI.

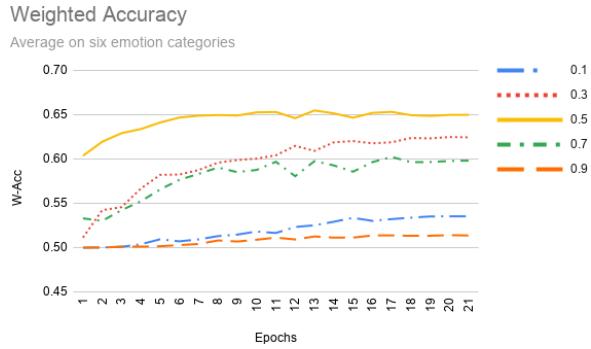


Figure 5: Trend lines of the weighted accuracy (W-Acc) score during training on the validation set of CMU-MOSEI.

## B Weighted F1 Analysis

In Figure 4 and 5, we show the trends of the weighted F1 (W-F1) and weighted accuracy (W-Acc) on the validation set of CMU-MOSEI during the training phase. The lines represent different threshold values as shown in the legend of each figure. The W-F1 is almost proportional to the threshold values and it is still very high when the threshold is 0.9 (i.e. most data samples are predicted to be negative). Moreover, when the threshold is large, the W-F1 keeps a high value starting from epoch 1. By contrast, the W-Acc score is more reliable. It ensures the model can also retrieve positive samples. We observe a similar phenomenon on all models. As a result, we think W-F1 is unsuitable as an evaluation metric on this dataset.

# All-in-One: A Deep Attentive Multi-task Learning Framework for Humour, Sarcasm, Offensive, Motivation, and Sentiment on Memes

Dushyant Singh Chauhan, Dhanush S R, Asif Ekbal and Pushpak Bhattacharyya

Department of Computer Science & Engineering

Indian Institute of Technology Patna

Patna, Bihar, India-801106

{1821CS17, dhanush.cs16, asif, pb}@iitp.ac.in

## Abstract

In this paper, we aim at learning the relationships and similarities of a variety of tasks, such as humour detection, sarcasm detection, offensive content detection, motivational content detection and sentiment analysis on a somewhat complicated form of information, *i.e.*, memes. We propose a multi-task, multi-modal deep learning framework to solve multiple tasks simultaneously. For multi-tasking, we propose two attention-like mechanisms *viz.*, Inter-task Relationship Module (*iTRM*) and Inter-class Relationship Module (*iCRM*). The main motivation of *iTRM* is to learn the relationship between the tasks to realize how they help each other. In contrast, *iCRM* develops relations between the different classes of tasks. Finally, representations from both the attentions are concatenated and shared across the five tasks (*i.e.*, humour, sarcasm, offensive, motivational, and sentiment) for multi-tasking. We use the recently released dataset in the Memotion Analysis task @ SemEval 2020, which consists of memes annotated for the classes as mentioned above. Empirical results on Memotion dataset show the efficacy of our proposed approach over the existing state-of-the-art systems (Baseline and SemEval 2020 winner). The evaluation also indicates that the proposed multi-task framework yields better performance over the single-task learning.

## 1 Introduction

The content and form of content shared on online social media platforms have changed rapidly over time. Currently, one of the most popular forms of media shared on such platforms is '*Memes*'. According to its definition from Oxford Dictionary, a meme is a piece of data, often in the form of images, text or videos that carry cultural information through an imitable phenomenon with a mimicked theme, that is shared (sometimes with slight modification) rapidly by internet users.

Every meme can be associated with five affect values, namely *humour* (Hu), *sarcastic* (Sar), *offensive* (Off), *motivational* (Mo), and *sentiment* (Sent). Hence, in a broad sense, memes can be categorized into four *intersecting* sets *viz.* humorous memes, sarcastic memes, offensive memes, and motivational memes.

**Humour** refers to the quality of being amusing or comic. Formally, humour is defined as the nature of experiences to induce laughter and provide amusement. Humorous memes are the most popular and widely used on social media platforms. An example for humorous memes is shown in Figure 1a.

**Sarcasm** is often used to convey thinly veiled disapproval humorously. A sarcastic meme is a meme where an incongruity exists between the intended meaning and the way it is expressed. These are generally used to express dissatisfaction or to veil insult through humour. As we can see in Figure 1a, the person on the right is made fun of, without explicitly expressing it, which is a typical example of a sarcastic meme.

**Offensive** content include a lot of insulting, derogatory terms. It is contrary to the moral sense or good. As social media expands, offensive language has become a huge headache to maintain sanity on social media. As memes are growing to become more and more popular, detecting offensive memes on such platforms is becoming an important and challenging task. Figure 1a, Figure 1c and Figure 1d are the instances of Offensive memes.

**Motivation** is derived from the word 'motive' which means needs or desires within the individuals. It is the process of stimulating people to actions to achieve their goals. By its definition, motivational memes are those that benefit a certain group of people to achieve their plans or goals. Motivation can be both either positive or negative.



Figure 1: Few examples from the *Memotion* dataset to show the inter-dependency between different tasks.

However, we usually consider motivation in a positive sense. Figure 1b is an excellent example for the positive motivation.

**Sentiment** analysis refers to the process of computationally identifying and categorizing opinions expressed in a piece of communication, especially to determine whether the writer’s attitude towards a particular topic, product, etc. is *positive*, *negative*, or *neutral*. This has been a very prominent and important task in Natural Language Processing. Sentiment analysis on memes refers to the task of systematically extracting its emotional tone in understanding the opinion expressed by the meme. Figure 1b is an example for positive sentiment towards the government and Figure 1c for negative sentiment towards *Ph.D.* in Electrical Engineering.

Generally, specific labels of one task have a strong relation to the other labels of sarcasm, offensive, humour or motivational tasks. Through proper representation, training, and evaluation, these relations can be modelled to help each other for better classification. For example, in Figure 1b, just by seeing text, the meme can be either sarcastic or motivational, but the image in the meme confirms that this has an overall positive sentiment and hence motivational. Similarly, in Figure 1c, knowing that the meme is sarcastic and has a negative sentiment makes it highly probable to being offensive.

As seen above, humorous, motivational, offensive, and sarcastic nature of the memes are closely related. Thus, a multi-task learning framework would be extremely beneficial in such scenarios. In this paper, we exploit these relationships and similarities in the tasks of humour detection, sarcasm detection, offensive content detection, motivational content detection, and sentiment in a multi-task manner. The main contributions and/or attributes are as follows: (a). We propose a multi-task multi-modal deep learning framework to leverage the util-

ity of each task to help each other in a multi-task framework; (b). We propose two attention mechanisms viz. *iTRM* and *iCRM* to better understand the relationship between the tasks and between the classes of tasks, respectively; and (c). We present the state-of-the-art results for meme prediction in the multi-modal scenario.

## 2 Related Work

Sentiment analysis and its related tasks, such as humour detection, sarcasm detection, and offensive content detection, are the topics of interest due to their needs in recent times. There has been a phenomenal growth in multi-modal information sources in social media, such as audio, video, and text. Multi-modal information analysis has attracted the attention of researchers and developers due to their complexity, and multi-tasking has been of keen interest in the field of affect analysis.

**Humour:** Early feature-based models attempt to solve humour include the models based on word overlap with jokes, presence of ambiguity, and word overlap with common idioms (Sjöbergh and Araki, 2007), human-centeredness, and negative polarity (Mihalcea and Pulman, 2007). Some of the recent multi-modal approaches include utilizing information from the various modalities, such as acoustic, visual, and text, using deep learning models (Bertero and Fung, 2016; Yang et al., 2019; Swamy et al., 2020). Yang et al. (2020) employs a paragraph decomposition technique coupled with fine-tuning BERT (Devlin et al., 2018) model for humour detection on three languages (Chinese, Spanish and Russian).

**Sarcasm:** Starting from the traditional approaches, such as rule-based methods (Veale and Hao, 2010), lexical features (Carvalho et al., 2009), and incongruity (Joshi et al., 2015) to all the way up to multi-modal deep learning techniques (Schi-

fanella et al., 2016), sarcasm detection has been showing its presence. Castro et al. (2019) created a multi-modal conversational dataset, MUStARD from the famous TV shows, and provided baseline SVM approaches for sarcasm detection. Recently, Chauhan et al. (2020) proposed a multi-task learning framework for multi-modal sarcasm, sentiment and emotion analysis to explore how sentiment and emotion helps sarcasm. The author used the MUStARD dataset and extended the MUStARD dataset with *sentiment* (implicit and explicit) and *emotion* (implicit and explicit) labels.

**Offensive:** Razavi et al. (2010) used a three-level classification model taking advantage of various features from statistical models and rule-based patterns and various dictionary-based features. Chen et al. (2012) proposed a feature-based Lexical Syntactic Feature (LSF) architecture to detect the offensive contents. Gomez et al. (2020) created a multi-modal hate-speech dataset from Twitter (*MMHS150K*) to introduce a deep-learning-based multi-modal Textual Kernels Model (TKM) and compare it with various existing deep learning architectures on the proposed MMHS150K dataset.

**Motivation:** Swieczkowska et al. (2020) proposes a novel chaining method of neural networks for identifying motivational texts where the output from one model is passed on to the second model.

**Sentiment:** An important task to leverage multi-modality information effectively is to combine them using various strategies. Mai et al. (2019) employs a hierarchical feature fusion strategy, *Divide, Conquer, and Combine* for affective computing. Chauhan et al. (2019) uses the Inter-modal Interaction Module (IIM) to combine information from a pair of modalities for multi-modal sentiment and emotion analysis. Some of the other techniques include a contextual inter-modal attention based framework for multi-modal sentiment classification (Ghosal et al., 2018; Akhtar et al., 2019).

**Multi-task:** Some of the early attempts to correlate the tasks like sarcasm, humour, and offensive statements include a features based classification using various syntactic and semantic features, such as frequency of words, the intensity of adverbs and adjectives, the gap between positive and negative terms, the structure of the sentence, synonyms and others (Barbieri and Saggion, 2014). More recently, Badlani et al. (2019) proposed a convolution-based model to extract the embedding by fine-tuning the same for the tasks of sentiment, sarcasm, humour,

and hate-speech and then concatenating these representations to be used in a sentiment classifier.

In our current work, we propose a multi-task multi-modal deep learning framework to simultaneously solve the tasks of sarcasm, humour, offensive, and motivational on memes. Further, to the best of our knowledge, this is the very first attempt at solving the multi-modal affect analysis on memes in a multi-task deep learning framework. We demonstrate through a detailed empirical evaluation that a multi-task learning framework can improve the performance of individual tasks over a single task learning framework.

### 3 Proposed Methodology

We propose an attention-based deep learning model to solve the problem of multi-task affect analysis of memes. The inputs to the model are the meme itself and the manually corrected text extracted through OCR. The overall architecture is depicted in Figure 2. The source code is available at <http://www.iitp.ac.in/~ai-nlp-ml/resources.html>.

#### 3.1 Input Layer:

We now describe the input features for our proposed model.

##### 3.1.1 Text Input

Given  $N$  number of samples, where each sample is associated with meme image and the corresponding text. Let us assume, in each sample, there are  $n_T$  number of words  $w_{1:n_T} = w_1, \dots, w_{n_T}$ , where  $w_j \in \mathbb{R}^{d_T}$ ,  $d_T = 768$ , and  $w_j$  is obtained using *BERT* (Devlin et al., 2018). The maximum number of words for  $i^{th}$  sample across the dataset is 189.

##### 3.1.2 Image Input

Image is the prime component of any meme and contains the majority of the information. To leverage this information effectively, feature vectors from average pooling layer (avgpool) of the ImageNet pre-trained *ResNet-152* (He et al., 2016) image classification model are extracted. Each image is first pre-processed by resizing to  $224 \times 224$  and then normalized. The extracted feature vector for image of  $i^{th}$  sample is represented by  $V_i \in \mathbb{R}^{d_v}$  and  $d_v = 2048$ .

### 3.2 Attention Modules

These vectors are concatenated and then passed through a set of four dense layers to obtain the vectors of equal length  $d$  represented by  $TV_t \in \mathbb{R}^d$ ,

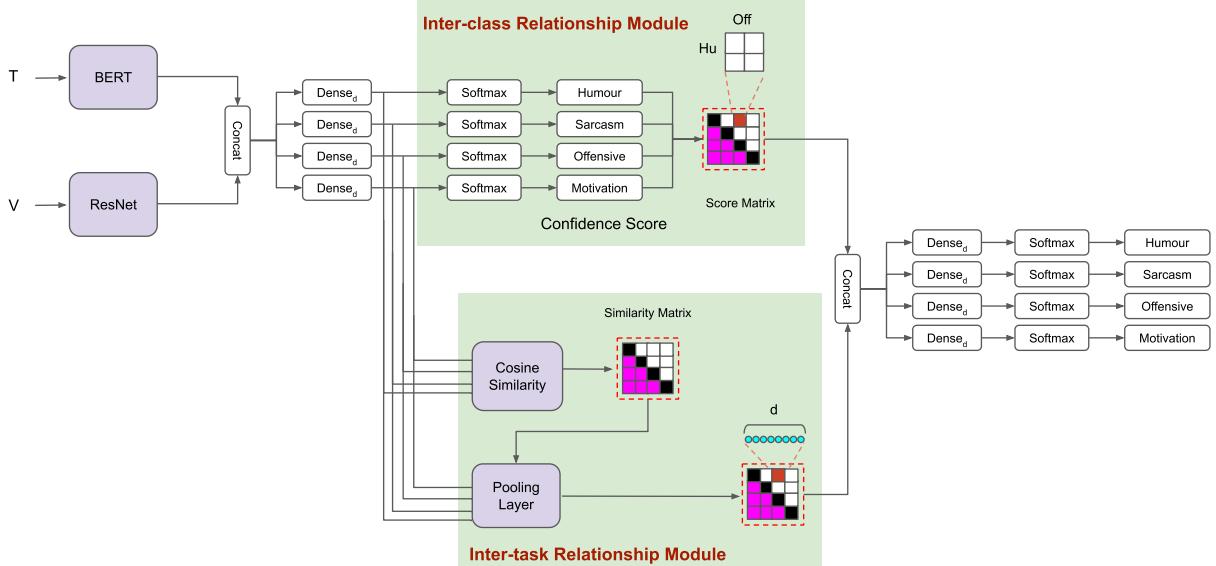


Figure 2: Overall architecture of the proposed multi-modal multi-task affect analysis framework for Memes. Here  $V$  refers to the *Meme Image* and  $T$  refers to the *text extracted from the Meme*.

where  $t$  is a task  $\in \{\text{humour, sarcasm, offensive, motivational}\}$ . These vectors are then passed through the Inter-class Relationship Module and Inter-task Relationship module. The output is then concatenated and passed through another set of four dense layers, and a layer of softmax is applied to obtain the final output.

### 3.2.1 Inter-class Relationship Module

This module is used to learn the relationship between the classes of all the tasks. This is done by passing  $TV_t$  through another dense layer and softmax (*confidence score*). For each task, we first group all the classes into two classes for the hierarchical classification of the sample. At this level, the sample is labelled with either positive or negative for all the tasks. For instance, a sample will be labelled as either sarcastic or not\_sarcastic for sarcasm tasks. A loss is back-propagated using these confidence scores for the corresponding tasks. This is done in order to control each dense layer so that it aligns with the respective tasks. Meanwhile, a dot-product of the softmax scores of each task is obtained and used to form the *Score Matrix*. This is then flattened and passed forward.

### 3.2.2 Inter-task Relationship Module

While the above module is used to find the correlation between the individual classes, this module is used to find the relationship between the different tasks in the model. This is done by initially finding the cosine-similarity between  $TV_t$  vectors. And a

pooling layer is used to collect information between the tasks and then normalized by the corresponding cosine-similarity score. The output from the pooling layer is then flattened and passed forward.

### 3.3 Output Unit

The flattened vectors from *iTRM* and *iCRM* are concatenated and then branched into four dense layers for each task. This is then forwarded through a softmax layer to obtain the final output for each task, and the loss is back-propagated to learn the parameters. In this layer, the information from both *iCRM* and *iTRM* modules will be leveraged and used to predict the final outcome. *Please note that, there are two sets of loss used in the model, one in the iCRM module and second at the end of Output Unit.*

## 4 Dataset

We perform experiments using the dataset released in the Memotion Analysis 1.0 @ SemEval 2020 Task (Sharma et al., 2020)<sup>1</sup>. This dataset consists of 6992 samples. Each sample consists of an image, corrected text extracted from the meme, and the five labels associated with the five tasks, *viz.*, *Humour, Sarcasm, Offensive, Motivational, and Overall Sentiment*. The distribution of the classes associated with each of the five tasks with label is shown in Table 1 and Table 2.

<sup>1</sup><https://competitions.codalab.org/competitions/20629>

Task	Classes	Count	RC (%)	T-A
Sent	very_negative	1033	17.34	$N_g$
	negative	3127	52.48	
	neutral	2201	36.94	
	positive	480	8.06	
	very_positive	151	2.53	$P_s$

Table 1: Dataset Distribution of Task-A, where  $RC$  and  $T-A$  denotes the relative count and abbreviation for labels of Task-A, respectively.

Task	Classes	Count	RC (%)	T-C	T-B
Hu	not_funny	1651	30.91	$N_f$	$N_h$
	funny	2452	45.91	$F_n$	
	very_funny	2238	41.90	$V_f$	
	hilarious	651	12.19	$H_r$	
Sar	not_sarcastic	1544	22.08	$N_s$	$S_r$
	general	3507	50.16	$G_r$	
	twisted_meaning	1547	22.13	$T_m$	
	very_twisted	394	5.64	$V_t$	
Off	not_offensive	2713	38.80	$N_o$	$O_f$
	slight	2592	37.07	$S_g$	
	very_offensive	1466	20.97	$V_o$	
	hateful_offensive	221	3.16	$H_o$	
Mo	not_motivational	4525	64.72	$N_m$	$M_o$
	motivational	2467	35.28	$M_o$	

Table 2: Dataset Distribution of Task-B and Task-C, where  $RC$ ,  $T-B$  and  $T-C$  denotes the relative count, abbreviation for labels of Task-B, and abbreviation for labels of Task-C respectively.

We address 5 multi-modal affective analysis problems, namely *humour classification*, *sarcasm classification*, *offensive classification*, *motivational classification*, and *sentiment classification*.

- A. Humour classification:** There are four classes associated with the humour task, namely not\_funny, funny, very\_funny, and hilarious, which are labelled as 0, 1, 2, and 3, respectively.
- B. Sarcasm classification:** There are four classes associated with the sarcasm task, namely not\_sarcastic, general, twisted\_meaning, and very\_twisted which are labelled as 0, 1, 2, and 3 respectively.
- C. Offensive classification:** There are four classes associated with the offensive task, namely not\_offensive, slight, very\_offensive, and hateful\_offensive which are labelled as 0, 1, 2, and 3, respectively.
- D. Motivational classification:** There are two classes associated with the motivational task,

namely not\_motivational and motivational, which are labelled as 0 and 1, respectively.

**E. Sentiment classification:** There are five classes associated with the sentiment task, namely very\_negative, negative, neutral, positive, and very\_positive, which are labelled as 0, 1, 2, 3, and 4, respectively.

## 5 Experimental setup

In accordance with the SemEval 2020 (Sharma et al., 2020), the project is organized into three sets of tasks<sup>2</sup>.

- **Task A: Sentiment Classification:** In this task, memes are classified into 3 classes viz., -1 (negative, very\_negative), 0 (neutral) and +1 (positive, very\_positive).
- **Task B: Binary-class Classification:** In this set of tasks, the memes are classified as follows (c.f. T-B in Table 2);
  1. **Humour** (funny, very\_funny, hilarious) and Non-humour (not\_funny).
  2. **Sarcasm** (general, twisted\_meaning, very\_twisted) and Non-sarcasm (non\_sarcastic)
  3. **Offensive** (slight, very\_offensive, hateful\_offensive) and Non-Offensive (not\_offensive), and
  4. **Motivational** (motivational) and Non-motivational (not\_motivational).

- **Task C: Multi-class Classification:** In this set of task, the original labels are used as described in the dataset (c.f. T-C in Table 2) for the tasks of Humour, Sarcasm, Offensive and Motivational.

Please note that, in Task A, as it is not a multi-task scenario, *iCRM* and *iTRM* are not applicable. For all the other sets of tasks, the entire network is shown in Figure 2.

We evaluate our proposed model on the multi-modal Memotion dataset. We perform *grid search* to find the optimal hyper-parameters (c.f. Table 3). Though we aim for a generic hyper-parameter configuration for all the experiments, in some cases, a different choice of the parameter has a significant effect. Therefore, we choose different parameters for a different set of experiments.

<sup>2</sup>[https://competitions.codalab.org/competitions/20629#learn\\_the\\_details-task-labels-format](https://competitions.codalab.org/competitions/20629#learn_the_details-task-labels-format)

Parameters	Task-A	Task-B	Task-C
Activations	<i>ReLU</i>		
Optimizer	<i>Adam (lr=0.001)</i>		
Output	<i>Softmax</i>		
Loss	<i>Categorical cross-entropy</i>		
Batch	16		
Epochs	30		
Dropout-p	0.3	0.5	0.7
#neurons(Dense)	50	200	200

Table 3: Model configurations

We implement our proposed model on the open source machine learning library PyTorch<sup>3</sup>. Hugging Face<sup>4</sup> library is used for BERT implementation. As the evaluation metric, we employ precision (P), recall (R), macro-F1 ( $M_a$ -F1), and micro-F1 ( $M_i$ -F1) for all the tasks *i.e.*, *humour, sarcasm, offensive, motivational, and sentiment*. We use *Adam* as an optimizer, *Softmax* as a classifier, and the *categorical cross-entropy* as a loss function for all the tasks.

## 6 Results and Analysis

We evaluate our proposed architecture with bi-modal inputs (*i.e.*, *text and visual*). We show the obtained results for Task-A (*i.e.*, *sentiment analysis*) in Table 4.

Labels	Task-A			
	P	R	$M_a$ -F1	$M_i$ -F1
Sentiment	36.99	35.70	35.81	50.58

Table 4: Memes: Sentiment Classification (Task A)

Task-B has four different tasks, *i.e.*, *humour, sarcasm, offensive, and sentiment* with binary-class labels (c.f. binary-class classification in Section 5). The results are shown in Table 5.

Labels	Task-B (Binary Classification)							
	STL				MTL			
	P	R	$M_a$ -F1	$M_i$ -F1	P	R	$M_a$ -F1	$M_i$ -F1
Hu	55.44	53.77	53.74	71.29	55.52	53.84	53.84	71.29
Sa	51.94	51.34	50.98	70.76	52.99	52.48	52.52	70.94
Of	52.33	52.19	52.13	56.28	51.35	51.37	51.36	54.10
Mo	53.56	53.49	53.51	57.18	55.86	56.44	56.12	57.44

Table 5: Memes: Single-task vs Multi-task (Task B)

Task-C has also four different tasks, *i.e.*, *humour, sarcasm, offensive, and sentiment* with multi-class labels (c.f. multi-class classification in Section 5). The results are shown in Table 6.

Labels	Task-C (Multi-class Classification)							
	STL				MTL			
	P	R	$M_a$ -F1	$M_i$ -F1	P	R	$M_a$ -F1	$M_i$ -F1
Hu	26.83	26.89	26.75	29.76	27.23	27.29	27.03	32.00
Sa	25.16	26.71	25.74	36.52	26.30	27.33	26.80	39.94
Of	27.21	27.30	26.93	35.30	25.05	26.04	25.53	35.94
Mo	53.32	52.89	52.65	58.46	54.14	53.31	53.72	59.79

Table 6: Memes: Single-task vs Multi-task (Task C)

In both the tasks B and C, we outline the comparison between the multi-task (MTL) and single-task (STL) learning frameworks in Table 5 and Table 6. We observe that MTL shows better performance over the STL setups.

For the offensive task, we find that STL performs better than MTL. We hypothesize that this is due to the model getting confused between the offensive and sarcastic (or humorous) memes. From Table 9, under Sarcasm, we can see that for the class  $V_t$ , MTL predicts a few samples as sarcastic, whereas in actuality it belongs to the other classes. However, we can see a decrease in performance for class  $H_o$  under Offensive. This is due to the lack of a larger dataset for the complex model to disambiguate the same. In the example, *BRB...GOT TO TAKE CARE OF SOME SH\*T IN UKRAIN* (c.f. Figure 1d), the actual set of labels are  $F_n, G_n, S_g, N_m$ . The predicted labels in STL are  $V_f, G_n, S_g, M_o$  and in MTL are  $V_f, T_m, V_o, M_o$ . This is supposed to be slightly offensive but got it confused with the sarcastic.

## 7 Comparative Analysis

We compare the results obtained in our proposed model against the baseline model and SemEval 2020 winner, which also made use of the same dataset. The comparative analysis is shown in Table 7. Our proposed multi-modal framework achieves the best macro-F1 of 35.8% (0.4%  $\uparrow$ ) and micro-F1 of 50.6% (1.9%  $\uparrow$ ) as compared to macro-F1 of 35.4% and micro-F1 of 48.7% of the state-of-the-art system (i.e., SemEval 2020 Winner) for Task-A. Similarly, for Task-B, we obtain the macro-F1 of 53.5% (1.7%  $\uparrow$ ) and micro-F1 of 63.4% (2.0%  $\uparrow$ ) as compared to the macro-F1 of 51.8% and micro-F1 of 61.4% of the state-of-the-art system, whereas for Task-C, we obtain the macro-F1 of 33.3% (1.1%  $\uparrow$ ) and micro-F1 of 41.9% (4.1%  $\uparrow$ ) as compared to the macro-F1 of 32.2% and micro-F1 of 37.8% of the state-of-the-art system.

It is evident from Table 5 and Table 6 that multi-task learning framework successfully leverages the

<sup>3</sup><https://pytorch.org/>

<sup>4</sup><https://github.com/huggingface/transformers>

Systems	Task A			Task B			Task C			
	M <sub>a</sub> -F1	M <sub>i</sub> -F1	M <sub>a</sub> -F1							
<b>Baseline</b>	21.76		30.77		50.02		56.86		30.08	
<b>SE'20 Winner</b>	35.46		48.72		51.83		61.44		32.24	
<b>Proposed</b>	<b>35.81</b>		<b>50.58</b>		<b>53.46</b>		<b>63.44</b>		<b>33.27</b>	
	<b>41.92</b>									

Table 7: Comparative Analysis of the proposed approach with recent state-of-the-art systems. Here, SE'20 denotes the SemEval 2020 winner, and 'Proposed' refers to the models described in the paper for the respective tasks.

Sentiment				Setups		Humour		Sarcasm		Offensive		Motivational	
	N <sub>g</sub>	N <sub>u</sub>	P <sub>s</sub>			N <sub>h</sub>	H <sub>m</sub>	N <sub>s</sub>	S <sub>r</sub>	N <sub>o</sub>	O <sub>f</sub>	N <sub>m</sub>	M <sub>o</sub>
N <sub>g</sub>	17	19	127	STL	N <sub>h</sub>	91	354	N <sub>s</sub>	S <sub>r</sub>	N <sub>o</sub>	O <sub>f</sub>	N <sub>m</sub>	M <sub>o</sub>
N <sub>u</sub>	25	170	399		H <sub>m</sub>	185	1248						
P <sub>s</sub>	58	290	763	MTL	N <sub>h</sub>	92	353	N <sub>s</sub>	S <sub>r</sub>	N <sub>o</sub>	O <sub>f</sub>	N <sub>m</sub>	M <sub>o</sub>
					H <sub>m</sub>	186	1247						

(a) Task-A

(b) Task-B

Table 8: Confusion Matrix for Task-A and Task-B (Refer Table 1 and Table 2 for Label definitions).

Setups		Humour				Sarcasm				Offensive				Motivational				
STL		N <sub>f</sub>	F <sub>n</sub>	V <sub>f</sub>	H <sub>r</sub>		N <sub>s</sub>	G <sub>r</sub>	T <sub>m</sub>	V <sub>t</sub>		N <sub>o</sub>	S <sub>g</sub>	V <sub>o</sub>	H <sub>o</sub>		N <sub>m</sub>	M <sub>o</sub>
	N <sub>f</sub>	122	143	130	50	N <sub>s</sub>	117	182	122	0	N <sub>o</sub>	254	307	111	35	N <sub>m</sub>	878	310
	F <sub>n</sub>	140	218	205	91	G <sub>r</sub>	234	427	276	0	S <sub>g</sub>	224	340	105	40			
	V <sub>f</sub>	129	201	193	82	T <sub>m</sub>	94	188	142	0	V <sub>o</sub>	109	198	62	18	M <sub>o</sub>	470	220
	H <sub>r</sub>	36	65	47	26	V <sub>t</sub>	19	52	25	0	H <sub>o</sub>	20	37	11	7			
MTL	N <sub>f</sub>	147	147	136	21	N <sub>s</sub>	125	206	87	3	N <sub>o</sub>	350	219	138	0	N <sub>m</sub>	924	264
	F <sub>n</sub>	173	240	208	33	G <sub>r</sub>	222	525	172	18	S <sub>g</sub>	330	250	129	0			
	V <sub>f</sub>	172	195	204	34	T <sub>m</sub>	112	210	100	2	V <sub>o</sub>	181	131	75	0	M <sub>o</sub>	491	199
	H <sub>r</sub>	51	70	43	10	V <sub>t</sub>	23	57	16	0	H <sub>o</sub>	43	22	10	0			

Table 9: Confusion Matrix for Task C (Refer Table 2 for Label definitions).

inter-dependence between all the tasks in improving the overall performance in comparison to the single-task learning. We also show the confusion matrices corresponding to each set of tasks in Table 8a, Table 8b, and Table 9, respectively.

## 8 Error Analysis

We perform error analysis (i.e. for Task-C) on the predictions of our proposed model. We take some utterances (c.f. Table 10) with corresponding image (c.f. Figure 3), where we show that *MTL* is predicting correct while *STL* is not able to predict the right labels.

We also present the attention heatmaps for *iCRM* and *iTRM* of the multi-task learning framework in Figure 4 and Figure 5, respectively. We take the fifth utterance from Table 10 (c.f. Figure 3e) to illustrate the heatmap. For *iCRM* (c.f. Figure 4), there are six matrices which show the inter-dependency between humour and sarcasm (*Hu-Sar*), humour and offensive (*Hu-Off*), humour and motivational (*Hu-Mo*), sarcasm and offensive (*Sar-Off*), sarcasm and motivational (*Sar-Mo*), and offensive and motivational (*Off-Mo*), respectively, where



Figure 3: Few examples for Human Error Analysis corresponding to Table 10.

the light shade to dark shade shows the amount of contributions in ascending sequence.

The main objective of *iCRM* is to develop the relationship between the classes of tasks. Figure 4 shows the established relationship between the tasks. We see the established relationship between the classes of tasks in Figure 4. For predicting the fifth utterance correctly in Table 10, humour and

	Utterances	STL				MTL			
		Hu	Sar	Off	Mo	Hu	Sar	Off	Mo
1	my name is giovanni giorgio but everybody calls me giorgio.	$N_f$	$G_r$	$N_o$	$N_m$	$V_f$	$T_m$	$V_o$	$M_o$
2	i'm in shape. unfortunately that shape is a potato	$V_f$	$N_s$	$N_o$	$M_o$	$F_n$	$G_r$	$S_g$	$N_m$
3	obama i'm coming after ur job as president memeshappen.Com	$F_n$	$G_r$	$N_o$	$N_m$	$V_f$	$T_m$	$V_o$	$M_o$
4	look at me I'm the captain now.	$V_f$	$T_m$	$V_o$	$M_o$	$F_n$	$G_r$	$S_g$	$N_m$
5	freshmen .000000000127 seconds after the bell mr. bean go zoom zoom.	$H_r$	$N_s$	$S_g$	$M_o$	$F_n$	$G_r$	$M_o$	$N_m$
6	sorry i was working.	$F_n$	$T_m$	$V_o$	$M_o$	$V_f$	$G_r$	$S_g$	$N_m$

Table 10: Comparison between multi-task learning and single-task learning frameworks .Few error cases where MTL framework performs better than the STL framework.

not sarcasm (Figure 4a), humour and not offensive (Figure 4b) etc. are helping each other.

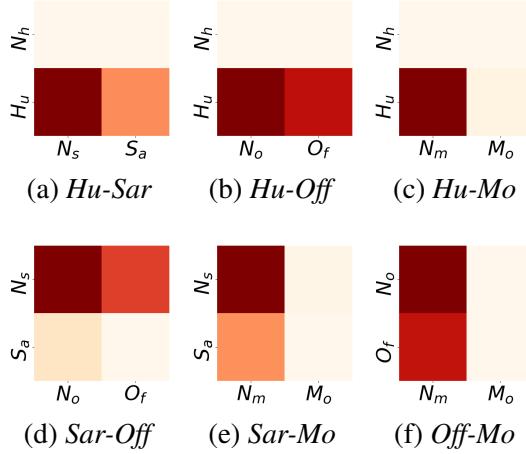


Figure 4: *iCRM* attention for Figure 3e under Task C

Similarly, the main objective of *iTRM* is to develop the relationship between the tasks. Figure 5 shows the established relationship between the tasks, and we see that attention put more weight on sarcasm and offensive pair while less weight on humour and sarcasm. It is clear from the definition of sarcasm and humour (c.f. Section 1) that both of them have a very different meaning when used in a sentence while the actual sentence looks similar. Hence sarcasm and humour is found not be helping each other.

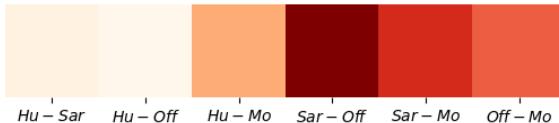


Figure 5: *iTRM* attention for Figure 3e under Task C

## 9 Conclusion and Future Work

In this paper, we have successfully established the concept of obtaining effective relationships

between inter-tasks and between inter-classes for multi-modal affect analysis. We have proposed a deep attentive multi-task learning framework which helps to obtain very effective inter-tasks and inter-classes relationship. To capture the interdependence, we have proposed two attention-like mechanisms *viz.*, Inter-task Relationship Module (*iTRM*) and Inter-class Relationship Module (*iCRM*). The main motivation of *iTRM* is to learn the relationship between the tasks, i.e. which task helps the other tasks. In contrast, *iCRM* develops the relations between the classes of tasks. We have evaluated our proposed approach on a recently published Memotion dataset. Experimental results suggest the efficacy of the proposed model over the existing state-of-the-art systems (Baseline and SemEval 2020 winner). The evaluation shows that the proposed multi-task framework yields better performance over single-task learning.

The dataset used for the experiments is relatively small for training an effective deep learning model and is heavily biased. Therefore, assembling a large, and more balance dataset with quality annotations is an important job. Moreover, the memes are a complicated form of data which includes both text and image that repeat over numerous memes (meme templates). Hence quality representation of memes for affect analysis is challenging future work.

## Acknowledgement

The research reported here is partially supported by SkyMap Global India Private Limited. Dushyant Singh Chauhan acknowledges the support of Prime Minister Research Fellowship (PMRF), Govt. of India. Asif Ekbal acknowledges the Young Faculty Research Fellowship (YFRF), supported by Visvesvaraya PhD scheme for Electronics and IT, Ministry of Electronics and Information Technology (Meit/8Y), Government of India, being implemented by Digital India Corporation (formerly Media Lab Asia).

## References

- Md Shad Akhtar, Dushyant Chauhan, Deepanway Ghosal, Soujanya Poria, Asif Ekbal, and Pushpak Bhattacharyya. 2019. Multi-task learning for multi-modal emotion recognition and sentiment analysis. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 370–379, Minneapolis, Minnesota. Association for Computational Linguistics.
- Rohan Badlani, Nishit Asnani, and Manan Rai. 2019. Disambiguating sentiment: An ensemble of humour, sarcasm, and hate speech features for sentiment classification. *W-NUT 2019*, page 337.
- Francesco Barbieri and Horacio Saggion. 2014. Automatic detection of irony and humour in twitter. In *ICCC*, pages 155–162.
- Dario Bertero and Pascale Fung. 2016. Multimodal deep neural nets for detecting humor in tv sitcoms. In *2016 IEEE Spoken Language Technology Workshop (SLT)*, pages 383–390. IEEE.
- Paula Carvalho, Luís Sarmento, Mário J Silva, and Eugénio De Oliveira. 2009. Clues for detecting irony in user-generated contents: oh...!! it's "so easy";-. In *Proceedings of the 1st international CIKM workshop on Topic-sentiment analysis for mass opinion*, pages 53–56.
- Santiago Castro, Devamanyu Hazarika, Verónica Pérez-Rosas, Roger Zimmermann, Rada Mihalcea, and Soujanya Poria. 2019. Towards multimodal sarcasm detection (an \_obviously\_ perfect paper). *arXiv preprint arXiv:1906.01815*.
- Dushyant Singh Chauhan, Md Shad Akhtar, Asif Ekbal, and Pushpak Bhattacharyya. 2019. Context-aware interactive attention for multi-modal sentiment and emotion analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5651–5661, Hong Kong, China. Association for Computational Linguistics.
- Dushyant Singh Chauhan, Dhanush S R, Asif Ekbal, and Pushpak Bhattacharyya. 2020. Sentiment and emotion help sarcasm? a multi-task learning framework for multi-modal sarcasm, sentiment and emotion analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4351–4360, Online. Association for Computational Linguistics.
- Ying Chen, Yilu Zhou, Sencun Zhu, and Heng Xu. 2012. Detecting offensive language in social media to protect adolescent online safety. In *2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Conference on Social Computing*, pages 71–80. IEEE.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
- Deepanway Ghosal, Md Shad Akhtar, Dushyant Singh Chauhan, Soujanya Poria, Asif Ekbal, and Pushpak Bhattacharyya. 2018. Contextual inter-modal attention for multi-modal sentiment analysis. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3454–3466, Brussels, Belgium. Association for Computational Linguistics.
- Raul Gomez, Jaume Gibert, Lluís Gomez, and Dimosthenis Karatzas. 2020. Exploring hate speech detection in multimodal publications. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 1470–1478.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.
- Aditya Joshi, Vinita Sharma, and Pushpak Bhattacharyya. 2015. Harnessing context incongruity for sarcasm detection. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 757–762.
- Sijie Mai, Haifeng Hu, and Songlong Xing. 2019. Divide, conquer and combine: Hierarchical feature fusion network with local and global perspectives for multimodal affective computing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 481–492.
- Rada Mihalcea and Stephen Pulman. 2007. Characterizing humour: An exploration of features in humorous texts. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 337–347. Springer.
- Amir H Razavi, Diana Inkpen, Sasha Uritsky, and Stan Matwin. 2010. Offensive language detection using multi-level classification. In *Canadian Conference on Artificial Intelligence*, pages 16–27. Springer.
- Rossano Schifanella, Paloma de Juan, Joel Tetreault, and Liangliang Cao. 2016. Detecting sarcasm in multimodal social platforms. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 1136–1145.
- Chhavi Sharma, Deepesh Bhageria, William Pak, Scott, Srinivas P Y K L, Amitava Das, Tanmoy Chakraborty, and Björn Gambäck. 2020. SemEval-2020 Task 8: Memotion Analysis-The Visuo-Lingual Metaphor! In *Proceedings of the 14th International Workshop on Semantic Evaluation (SemEval-2020)*, Barcelona, Spain. Association for Computational Linguistics.

Jonas Sjöbergh and Kenji Araki. 2007. Recognizing humor without recognizing meaning. In *International Workshop on Fuzzy Logic and Applications*, pages 469–476. Springer.

Steve Durairaj Swamy, Shubham Laddha, Basil Abdus-salam, Debsayan Datta, and Anupam Jamatia. 2020. Nit-agartala-nlp-team at semeval-2020 task 8: Building multimodal classifiers to tackle internet humor. *arXiv preprint arXiv:2005.06943*.

Patrycja Świeczkowska, Rafal Rzepka, and Kenji Araki. 2020. Stepwise noise elimination for better motivational and advisory texts classification. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, 24(1):156–168.

Tony Veale and Yanfen Hao. 2010. Detecting ironic intent in creative comparisons. In *ECAI*, volume 215, pages 765–770.

Hao Yang, Yao Deng, Minghan Wang, Ying Qin, and Shiliang Sun. 2020. Humor detection based on paragraph decomposition and bert fine-tuning.

Zixiaofan Yang, Lin Ai, and Julia Hirschberg. 2019. Multimodal indicators of humor in videos. In *2019 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, pages 538–543. IEEE.

# Identifying Implicit Quotes for Unsupervised Extractive Summarization of Conversations

Ryuji Kano<sup>†‡</sup> Yasuhide Miura<sup>†</sup> Tomoki Taniguchi<sup>†</sup> Tomoko Ohkuma<sup>†</sup>

<sup>†</sup>Fuji Xerox Co., Ltd.

<sup>‡</sup>Institute of Innovative Research, Tokyo Institute of Technology

{kano.ryuji, yasuhide.miura, taniguchi.tomoki, ohkuma.tomoko}  
@fujixerox.co.jp

## Abstract

We propose Implicit Quote Extractor, an end-to-end unsupervised extractive neural summarization model for conversational texts. When we reply to posts, quotes are used to highlight important part of texts. We aim to extract quoted sentences as summaries. Most replies do not explicitly include quotes, so it is difficult to use quotes as supervision. However, even if it is not explicitly shown, replies always refer to certain parts of texts; we call them implicit quotes. Implicit Quote Extractor aims to extract implicit quotes as summaries. The training task of the model is to predict whether a reply candidate is a true reply to a post. For prediction, the model has to choose a few sentences from the post. To predict accurately, the model learns to extract sentences that replies frequently refer to. We evaluate our model on two email datasets and one social media dataset, and confirm that our model is useful for extractive summarization. We further discuss two topics; one is whether quote extraction is an important factor for summarization, and the other is whether our model can capture salient sentences that conventional methods cannot.

## 1 Introduction

As the amount of information exchanged via online conversations is growing rapidly, automated summarization of conversations is in demand. Neural-network-based models have achieved great performance on supervised summarization, but its application to unsupervised summarization is not sufficiently explored. Supervised summarization requires tens of thousands of human-annotated summaries. Because it is not realistic to prepare such large datasets for every domain, there is a growing requirement for unsupervised methods.

Previous research proposed diverse methods of unsupervised summarization. Graph-centrality

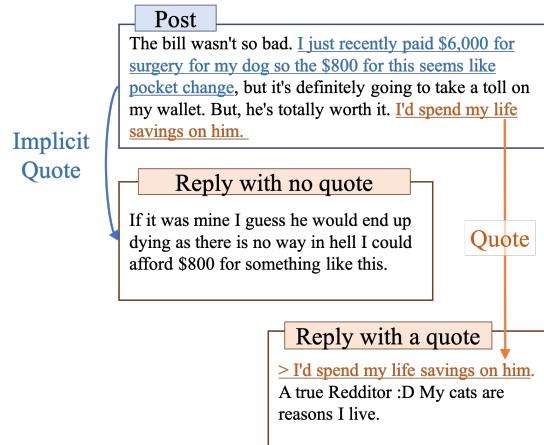


Figure 1: Example of a post and a reply with a quote and a reply with no quote. Implicit quote is the part of post that reply refers to, but not explicitly shown in the reply.

based on the similarity of sentences (Mihalcea and Tarau, 2004; Erkan and Radev, 2004; Zheng and Lapata, 2019) has long been a strong feature for unsupervised summarization, and is also used to summarize conversations (Mehdad et al., 2014; Shang et al., 2018). Apart from centrality, centroid of vectors (Gholipour Ghalandari, 2017), Kullback-Leibler divergence (Haghghi and Vanderwende, 2009), reconstruction loss (He et al., 2012; Liu et al., 2015; Ma et al., 2016), and path scores of word graphs (Mehdad et al., 2014; Shang et al., 2018), are leveraged for summarization.

The premise of these methods is that important topics appear frequently in a document. Therefore, if important topics appear only a few times, these methods fail to capture salient sentences. For more accurate summarization, relying solely on the frequency is not sufficient and we need to focus on other aspects of texts.

As an alternative aspect, we propose “the probability of being quoted”. When one replies to an email or a post, a quote is used to highlight the

important parts of the text; an example is shown in Figure 1. The reply on the bottom includes a quote, which generally starts with a symbol “>”. If we can predict quoted parts, we can extract important sentences irrespective of how frequently the same topic appears in the text. Thus, we aim to extract quotes as summaries.

Previous research assigned weights to words that appear in quotes, and improved the centroid-based summarization (Carenini et al., 2007; Oya and Carenini, 2014). However, most replies do not include quotes, so it is difficult to use quotes as the training labels of neural models. We propose a model that can be trained without explicit labels of quotes. The model is Implicit Quote Extractor (IQE). As shown in Figure 1, implicit quotes are sentences of posts that are not explicitly quoted in replies, but are those the replies most likely refer to. The aim of our model is to extract these implicit quotes for extractive summarization.

We use pairs of a post and reply candidate to train the model. The training task of the model is to predict if a reply candidate is an actual reply to the post. IQE extracts a few sentences of the post as a feature for prediction. To predict accurately, IQE has to extract sentences that replies frequently refer to. Summaries should not depend on replies, so IQE does not use reply features to extract sentences. The model requires replies only during the training and not during the evaluation.

We evaluate our model with two datasets of Enron mail (Loza et al., 2014), corporate and private mails, and verify that our model outperforms baseline models. We also evaluated our model with Reddit TIFU dataset (Kim et al., 2019) and achieved results competitive with those of the baseline models.

Our model is based on a hypothesis that the ability of extracting quotes leads to a good result. Using the Reddit dataset where quotes are abundant, we obtain results that supports the hypothesis. Furthermore, we both quantitatively and qualitatively analyzed that our model can capture salient sentences that conventional frequency-based methods cannot. The contributions of our research are as follows:

- We verified that “the possibility of being quoted” is useful for summarization, and demonstrated that it reflects an important aspect of saliency that conventional methods do not.

- We proposed an unsupervised extractive neural summarization model, Implicit Quote Extractor (IQE), and demonstrated that the model outperformed or achieved results competitive to baseline models on two mail datasets and a Reddit dataset.
- Using the Reddit dataset, we verified that quote extraction leads to a high performance of summarization.

## 2 Related Works

Summarization methods can be roughly grouped into two methods: extractive summarization and abstractive summarization. Most unsupervised summarization methods proposed are extractive methods. Despite the rise of neural networks, conventional non-neural methods are still powerful in the field of unsupervised extractive summarization.

The graph-centrality-based method (Mihalcea and Tarau, 2004; Erkan and Radev, 2004; Zheng and Lapata, 2019) and centroid-based method (Gholipour Ghalandari, 2017) have been major methods in this field. Other models use reconstruction loss (He et al., 2012; Liu et al., 2015; Ma et al., 2016), Kullback-Leibler divergence (Haghghi and Vanderwende, 2009) or path score calculation (Mehdad et al., 2014; Shang et al., 2018) based on multi-sentence compression algorithm (Filippova, 2010). These methods assume that important topics appear frequently in a document, but our model focuses on a different aspect of texts: the probability of being quoted. That is, our model can extract salient sentences that conventional methods fail to.

A few neural-network-based unsupervised extractive summarization methods were proposed (Kågebäck et al., 2014; Yin and Pei, 2015; Ma et al., 2016). However, these methods use pretrained neural network models as a feature extractor, whereas we propose an end-to-end neural extractive summarization model.

As for end-to-end unsupervised neural models, a few abstractive models have been proposed. For sentence compression, Fevry and Phang (2018) employed the task to reorder the shuffled word order of sentences. Baziotis et al. (2019) employed the reconstruction task of the original sentence from a compressed one. For review abstractive summarization, Isonuma et al. (2019) revealed parent nodes of tree structures induce summaries, Chu and Liu

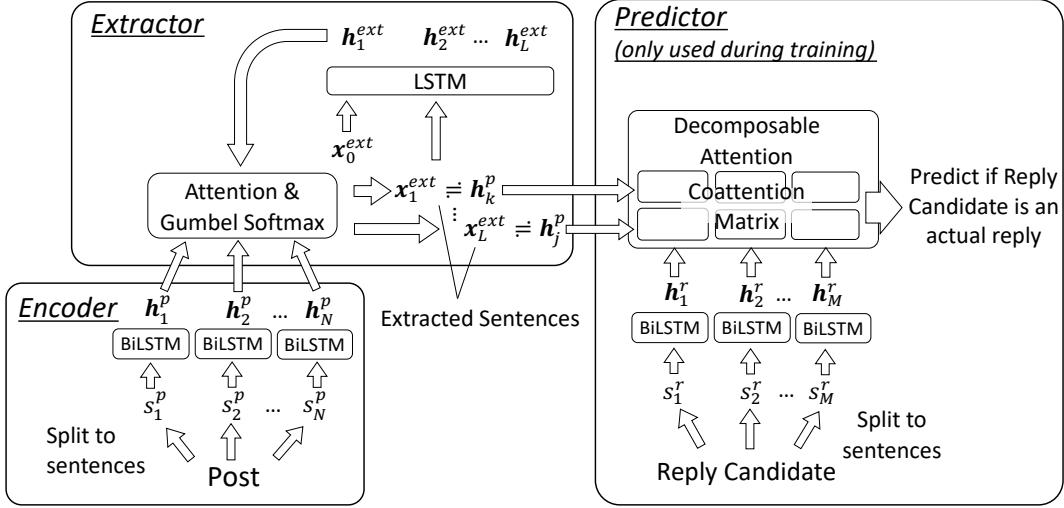


Figure 2: Description of our model, Implicit Quote Extractor (IQE). The Extractor extracts sentences and uses them as summaries.  $k$  and  $j$  are indices of the extracted sentences.

(2019) generated summaries from mean vectors of review vectors, and Amplayo and Lapata (2020) employed the prior distribution of Variational Auto-Encoder to induce summaries. Another research employed a task to reconstruct masked sentences for summarization (Laban et al., 2020).

Research on the summarization of online conversations such as mail, chat, social media, and online discussion fora has been conducted for a long time. Despite the rise of neural summarization models, most research on conversation summarization is based on non-neural models. A few used path scores of word graphs (Mehdad et al., 2014; Shang et al., 2018). Dialogue act classification is a classification task that classifies sentences depending on what their functions are (e.g.: questions, answers, greetings), and has also been applied for summarization (Bhatia et al., 2014; Oya and Carenini, 2014).

Quotes are also important factors of summarization. When we reply to a post or an email and when we want to emphasize a certain part of it, we quote the original text. A few studies used these quotes as features for summarization. Some previous work (Carenini et al., 2007; Oya and Carenini, 2014) assigned weights to words that appeared in quotes, and improved the conventional centroid-based methods. The previous research used quotes as auxiliary features. In our research, we solely focus on quotes, and do not directly use quotes as supervision; rather, we aim to extract implicit quotes.

### 3 Model

We propose Implicit Quote Extractor (IQE), an unsupervised extractive summarization model. Figure 2 shows the structure of the model. The inputs to the model during training are a post and reply candidate. A reply candidate can be either a true or a false reply to the post. The training task of the model is to predict whether a reply candidate is true or not.

The model comprises an Encoder, an Extractor, and a Predictor. The Encoder computes features of posts, the Extractor extracts sentences of a post to use for prediction, and the Predictor predicts whether a reply candidate is an actual reply or not. We describe each component below.

**Encoder** The Encoder computes features of posts. First, the post is split into  $N$  sentences  $\{s_1^p, s_2^p, \dots, s_N^p\}$ . Each sentence  $s_i^p$  comprises  $K_i$  words  $W_i^p = \{w_{i1}^p, w_{i2}^p, \dots, w_{iK_i}^p\}$ . Words are embedded to continuous vectors  $X_i^p = \{x_{i1}^p, x_{i2}^p, \dots, x_{iK_i}^p\}$  through word embedding layers. We compute the features of each sentence  $h_i^p$  by inputting embedded vectors to Bidirectional Long Short-Term Memory (BiLSTM) and concatenating the last two hidden layers:

$$h_i^p = \text{BiLSTM}(X_i^p) \quad (1)$$

**Extractor** The Extractor extracts a few sentences of a post for prediction. For accurate prediction, the Extractor learns to extract sentences that replies frequently refer to. Note that the Extractor does not use reply features for extraction. This is because

summaries should not depend on replies. IQE requires replies only during the training and can induce summaries without replies during the evaluation.

We employ LSTM to sequentially compute features on the Extractor. We set the mean vector of the sentence features of the Encoder  $\mathbf{h}_i^p$  as the initial hidden state of the Extractor  $\mathbf{h}_0^{ext}$ .

$$\mathbf{h}_0^{ext} = \frac{1}{N} \sum_{i=1}^N \mathbf{h}_i^p \quad (2)$$

The Extractor computes attention weights using the hidden states of the Extractor  $\mathbf{h}_t^{ext}$  and the sentence features  $\mathbf{h}_i^p$  computed on the Encoder. The sentence with the highest attention weight is extracted. During the training, we use Gumbel Softmax (Jang et al., 2017) to make this discrete process differentiable. By adding Gumbel noise  $g$  using noise  $u$  from a uniform distribution, the attention weights  $a$  become a one-hot vector. The discretized attention weights  $\alpha$  are computed as follows:

$$u_i \sim \text{Uniform}(0, 1) \quad (3)$$

$$g_i = -\log(-\log u_i) \quad (4)$$

$$a_{ti} = \mathbf{c}^T \tanh(\mathbf{h}_t^{ext} + \mathbf{h}_i^p) \quad (5)$$

$$\pi_{ti} = \frac{\exp a_{ti}}{\sum_{k=1}^N \exp a_{tk}} \quad (6)$$

$$\alpha_{ti} = \frac{\exp(\log \pi_{ti} + g_i)/\tau}{\sum_{k=1}^N \exp(\log \pi_{tk} + g_k)/\tau} \quad (7)$$

$c$  is a parameter vector, and the temperature  $\tau$  is set to 0.1. We input the linear sum of the attention weights  $\alpha$  and the sentence vectors  $\mathbf{h}_i^p$  to LSTM and update the hidden state of the Extractor. We repeat this step  $L$  times.

$$\mathbf{x}_t^{ext} = \sum_{i=1}^N \alpha_{ti} \mathbf{h}_i^p \quad (1 \leq t \leq L) \quad (8)$$

$$\mathbf{h}_{t+1}^{ext} = \text{LSTM}(\mathbf{x}_t^{ext}) \quad (0 \leq t \leq L-1) \quad (9)$$

The initial input vector  $\mathbf{x}_0^{ext}$  of the Extractor is a parameter, and  $L$  is defined by a user depending on the number of sentences required for a summary.

**Predictor** Then, using only the extracted sentences and a reply candidate, the Predictor predicts whether the candidate is an actual reply or not. We labeled actual replies as positive, and randomly sampled posts as negative. Suppose a reply candidate  $R = \{s_1^r, s_2^r, \dots, s_M^r\}$  has  $M$  sentences.

Sentence vectors  $\{\mathbf{h}_j^r\}$  of each sentence  $\{s_j^r\}$  on the reply are computed similarly to the equation 1. To compute the relation between the post and the reply candidate, we employ Decomposable Attention (Parikh et al., 2016).

From this architecture, we obtain the probability of binary-classification  $y$  through the sigmoid function.

$$y = \text{sigmoid}(\text{DA}(\mathbf{x}_1^{ext}, \dots, \mathbf{x}_{L-1}^{ext}, \mathbf{h}_1^r, \dots, \mathbf{h}_M^r)) \quad (10)$$

where DA denotes Decomposable Attention. The detail of the computation is described in Appendix A.1. Decomposable Attention.

The loss of this classification  $\mathcal{L}_{rep}$  is obtained by cross entropy as follows where  $t_{rep}$  is 1 when a reply candidate is an actual reply, and otherwise 0.

$$\mathcal{L}_{rep} = -t_{rep} \log y - (1 - t_{rep}) \log(1 - y) \quad (11)$$

**Reranking** As we mentioned in the Introduction, we are seeking for a criterion that is different from conventional methods. To take advantage of our method and conventional methods, we employ reranking; we simply reorder summaries (3 sentences) extracted by our model based on the ranking of TextRank (Mihalcea and Tarau, 2004).

## 4 Experiment

We train and evaluate the model on two domains of datasets. One is a mail dataset, and the other is a dataset from the social media platform, Reddit.

### 4.1 Mail Dataset

We use Avocado collection<sup>1</sup> for the training. The Avocado collection is a public dataset that comprises emails obtained from 279 custodians of a defunct information technology company. From this dataset, we use post-and-reply pairs to train our model. We exclude pairs where the number of words in a post or a reply is smaller than 50 or 25. After the preprocessing, we have 56,174 pairs. We labeled a pair with an actual reply as positive and a pair with a wrong reply that is randomly sampled from the whole dataset as negative. The number of positive labels and negative labels are equal. Therefore, we have 112,348 pairs in total.

For evaluation, we employ the Enron Summarization dataset (Loza et al., 2014). This dataset

<sup>1</sup><https://catalog.ldc.upenn.edu/LDC2015T03>

Data	Sample size	Summary			Source		
		# of references	# of sentences	# of words	# of sentences	# of words	# of words per sentence
ECS	109	2	4.7	78.0	11.0	179.4	16.3
EPS	103	2	5.8	88.0	19.3	217.1	11.2
tldr	3000	1	1.3	19.7	15.1	311.9	20.7

Table 1: Overview of the evaluation datasets.

has two types of evaluation datasets: ECS (Enron Corporate Single) and EPS (Enron Personal Single). An overview of these datasets is summarized in Table 1. Because the evaluation datasets do not have validation datasets, we use the ECS dataset as a validation dataset for the EPS dataset, and vice versa. We use the validation datasets to decide which model to use for the evaluation.

## 4.2 Reddit TIFU Dataset

The Reddit TIFU dataset (Kim et al., 2019) is a dataset that leverages tldr tags for the summarization task, which is the abbreviation of “too long didn’t read”. On the discussion forum Reddit TIFU, users post a tldr along with the post. tldr briefly explains what is written in the original post and thus can be regarded as a summary. We preprocess the TIFU dataset similarly as the mail datasets. Because the TIFU dataset does not include replies, we collected replies of the posts included in the TIFU dataset using praw<sup>2</sup>. As a consequence, we obtained 183,500 correct pairs of posts and replies and the same number of wrong pairs. We use that 367,000 pairs of posts and replies as the training dataset. We use 3,000 posts and tldrs that are not included in the training dataset as the validation dataset, and the same number of posts and tldrs as the evaluation dataset. An overview of the TIFU evaluation dataset is also summarized in Table 1.

## 4.3 Training

The dimensions of the embedding layers and hidden layers of the LSTM are 100. The size of the vocabulary is set to 30,000. We tokenize each email or post into sentences and each sentence into words using the nltk tokenizer<sup>3</sup>. The upper limit of the number of sentences is set to 30, and that of words in each sentence is set to 200. The epoch size is 10, and we use Adam (Kingma and Ba, 2015) as an optimizer.

In the first few epochs, we do not use the Extractor; all the post sentences are used for the prediction

of post-reply relations. This is to train the Extractor and the Predictor efficiently. The Extractor learns to extract proper sentences and the Predictor learns to predict the relation between a post and a reply candidate. Models with several components generally achieve better results if each component is pretrained separately (Hashimoto et al., 2017). Thus, we train the Predictor in the first few epochs before training the Extractor. We set this threshold as 4.

During training,  $L$ , the number of sentences the Extractor extracts is randomly set from 1 to 4, so that the model can extract an arbitrary number of sentences. We replace the named entities on the text data with tags (person, location, and organization) using the Stanford Named Entity Recognizer (NER)<sup>4</sup>, to prevent the model from simply using named entities as a hint for the prediction. We pre-train word embeddings of the model with Skipgram, using the same data as the training. We conduct the same experiment five times and use the average of the results to mitigate the effect of randomness rooting in initialization and optimization.

## 4.4 Evaluation

In the evaluation phase, we only use the Encoder and Extractor and do not use the Predictor. Each model extracts 3 sentences as a summary. Following previous work, we report the average F1 of ROUGE-1, ROUGE-2, and ROUGE-L for the evaluation (Lin, 2004). We use the first 20, 40, and 60 words of the extracted sentences. For ROUGE computation, we use ROUGE 2.0 (Ganesan, 2015). As a validation metric, we use an average of ROUGE-1-F, ROUGE-2-F, and ROUGE-L-F.

## 4.5 Baseline

As baseline models, we employ TextRank (Mihalcea and Tarau, 2004), LexRank (Erkan and Radev, 2004), KLSum (Haghghi and Vanderwende, 2009), PacSum (Zheng and Lapata, 2019), Lead, and Random.

TextRank and LexRank are graph-centrality based methods that have long been considered as

<sup>2</sup><https://praw.readthedocs.io/>

<sup>3</sup><https://www.nltk.org>

<sup>4</sup><https://nlp.stanford.edu/software/CRF-NER.shtml>

Model	ROUGE-1-F			ROUGE-2-F			ROUGE-L-F		
	# of words			# of words			# of words		
	20	40	60	20	40	60	20	40	60
Lead	0.217	0.351	0.413	0.115	0.198	0.240	0.212	0.290	0.321
TextRank	0.231	0.365	0.434	0.123	0.199	0.243	<u>0.223</u>	<u>0.294</u>	<b>0.336</b>
LexRank	0.234	0.359	0.423	0.127	0.199	0.240	0.220	0.290	0.323
Random	0.193	0.317	0.365	0.089	0.163	0.190	0.199	0.285	0.303
KLSum	0.235	0.344	0.383	0.125	0.183	0.204	0.220	0.273	0.303
PacSum	0.230	0.367	0.435	0.125	<b>0.211</b>	<b>0.256</b>	0.220	0.287	0.326
IQETextRank	0.213	0.336	0.394	0.104	0.172	0.208	0.211	0.287	0.315
IQE	<u>0.241</u>	<b>0.374</b>	<b>0.445</b>	<u>0.130</u>	0.206	<u>0.251</u>	0.220	0.292	<u>0.333</u>
IQE + reranking	<b>0.242</b>	<b>0.374</b>	0.443	<b>0.131</b>	0.207	0.246	<b>0.227</b>	<b>0.298</b>	0.332

Table 2: Results on ECS data. The best results are bolded and the second best results are underlined.

Model	ROUGE-1-F			ROUGE-2-F			ROUGE-L-F		
	# of words			# of words			# of words		
	20	40	60	20	40	60	20	40	60
Lead	0.128	0.204	0.230	0.045	0.084	0.099	0.150	0.208	0.221
TextRank	0.172	0.272	0.317	0.080	0.129	0.151	0.185	0.260	0.290
LexRank	0.161	0.254	0.299	0.068	0.113	0.136	0.173	0.245	0.275
Random	0.144	0.213	0.238	0.058	0.086	0.099	0.158	0.213	0.232
KLSum	<b>0.191</b>	0.287	0.321	<b>0.093</b>	0.141	0.153	0.184	0.254	0.277
PacSum	0.179	0.275	0.330	0.082	0.127	0.151	0.171	0.250	0.287
IQETextRank	0.158	0.252	0.291	0.069	0.115	0.136	0.169	0.242	0.268
IQE	<u>0.189</u>	<b>0.292</b>	<b>0.342</b>	0.091	<b>0.143</b>	<b>0.168</b>	<b>0.189</b>	<b>0.268</b>	<b>0.302</b>
IQE + reranking	0.185	0.290	0.340	0.087	0.138	<u>0.164</u>	<b>0.189</b>	0.264	0.299

Table 3: Results on EPS data. The best results are bolded and the second best results are underlined.

strong methods for unsupervised summarization. PacSum is an improved model of TextRank, which harnesses the position of sentences as a feature. KLSum employs the Kullback–Leibler divergence to constrain extracted sentences and the source text to have the similar word distribution. Lead is a simple method that extracts the first few sentences from the source text but is considered as a strong baseline for the summarization of news articles. PacSum and LexRank leverage idf. We compute idf using the validation data.

As another baseline, we employ IQETextRank; the TextRank model that leverages cosine similarities of sentence vectors of IQE’s Encoder as similarities between sentences. This is added to verify that the success of our model is not only because our model uses neural networks.

## 5 Results and Discussion

Experimental results for each evaluation dataset are listed in Table 2, 3 and 4. Our model outperforms baseline models on the mail datasets (ECS and EPS) in most metrics. On Reddit TIFU dataset, IQE with reranking outperforms most baseline models except TextRank. Reranking improves the accuracy on ECS and TIFU but not on EPS. PacSum significantly outperformed TextRank on

the news article dataset (Zheng and Lapata, 2019) but does not work well on our datasets where the sentence position is not an important factor. IQETextRank performed worse than IQE with the mail datasets. This indicates that the performance of our model does not result from the use of neural networks.

Our model outperforms the baseline models more with the EPS dataset than the ECS dataset. The overview of the datasets in Table 1 explains the reason. The average number of words each sentence has is smaller in EPS. Baseline models such as LexRank and TextRank compute similarity of sentences using the co-occurrence of words. Thus, if the lengths of sentences are short, it fails to build decent co-occurrence networks and to capture the saliency of the sentences. IQE did not outperform TextRank on TIFU dataset. It is conceivable that Reddit users are less likely to refer to important topics on the post, given that anyone can reply.

### 5.1 The Performance of Summarization and Quote Extraction

Our model performed well on the Mail datasets but two questions remain unclear. First, because we did not use quotes as supervision, it is not clear how well our model extracts quotes. Second, following

Model	ROUGE-1-F			ROUGE-2-F			ROUGE-L-F		
	# of words			# of words			# of words		
	20	40	60	20	40	60	20	40	60
Lead	0.128	0.150	0.149	0.017	0.023	0.024	0.107	0.122	0.125
TextRank	<b>0.161</b>	<u>0.179</u>	<u>0.173</u>	<b>0.027</b>	<u>0.034</u>	<u>0.035</u>	<b>0.126</b>	<u>0.140</u>	<u>0.142</u>
LexRank	0.149	0.165	0.163	0.021	0.026	0.029	0.119	0.131	0.134
Random	0.136	0.156	0.158	0.018	0.024	0.026	0.112	0.128	0.131
KL-Sum	0.142	0.159	0.157	0.020	0.026	0.029	0.115	0.127	0.131
PacSum	0.143	0.161	0.161	0.021	0.026	0.028	0.117	0.132	0.135
IQETextRank	0.152	0.169	0.166	0.023	0.030	0.032	0.122	0.136	0.139
IQE	0.153	0.172	0.169	0.024	0.031	0.033	0.122	0.136	<u>0.139</u>
IQE + reranking	<b>0.161</b>	<u>0.177</u>	<u>0.171</u>	<u>0.026</u>	<u>0.033</u>	<u>0.034</u>	<b>0.126</b>	<u>0.138</u>	<u>0.139</u>

Table 4: Results on TIFU tldr data. The best results are bolded and the second best results are underlined.

Model	MRR
LexRank	0.094
TextRank	0.109
Random	0.081
IQE	<b>0.135</b>

Table 5: Ability of extracting quotes.

Model	ROUGE-1-F	ROUGE-2-F	ROUGE-L-F
IQEquote	<b>0.184</b>	<b>0.030</b>	<b>0.126</b>
IQEnonquote	0.168	0.020	0.118

Table 6: ROUGE scores of extracted sentences that coincide with quote (IQE-quote) and that does not coincide with quotes (IQEnonquote). The ROUGE scores become higher when IQE succeeded in extracting quotes.

Carenini’s work (Carenini et al., 2007; Oya and Carenini, 2014), we assumed quotes were useful for summarization but it is not clear whether the quote extraction leads to better results of summarization. To answer these questions, we conduct two experiments.

For the experiments, we use the Reddit TIFU dataset and replies extracted via praw as described in 4.2. From the dataset, we extract replies that contain quotes, which start with the symbol “>”. In total, 1,969 posts have replies that include quotes. We label sentences of the posts that are quoted by the replies and verify how accurately our model can extract the quoted sentences.

**How well our model extracts quotes?** To assess the ability of quote extraction, we regard the extraction of quotes as an information retrieval task and evaluate with Mean Reciprocal Rank (MRR). We compute MRR as follows.

$$\text{MRR} = \begin{cases} \frac{1}{R(q)} & (R(q) \leq 4) \\ 0 & (R(q) > 4) \end{cases} \quad (12)$$

The function  $R$  denotes the rank of the saliency scores a model computes; our model does not compute the scores but sequentially extracts sentences, and the order is regarded as the rank here. If a model extracts quotes as salient sentences, the rank becomes higher. Therefore, the MRR in our study indicates the capability of a model to extract quotes. As explained in the section 4.3, we trained our model to extract up to four sentences. Thus we set

the threshold at four; if  $R(q)$  is larger than 4 we set MRR 0. For each data, we compute MRR and use the mean value as a result. Table 5 shows the results. IQE is more likely to extract quotes than TextRank, LexRank and Random.

**Does extracting quotes lead to good summarization?** Next, we validate whether the ROUGE scores become better when our model succeeded in extracting quotes. We compute ROUGE scores when our model succeeds or fails in quote extraction (which means when MRR equals 1 or otherwise). IQEquote indicates the data where the extracted sentence coincides with a quote, and IQEnonquote vice versa. The result in the Table 6 shows ROUGE scores are higher when the extracted sentence coincides with a quote. The results of the two analyses support the claim that our model is more likely to extract quotes and that the ability of extracting quotes leads to better summarization.

## 5.2 Ablation Tests

**Effect of replacing named entities** As explained in the section 4.3, our models shown in Tables 2, 3 and 4 all use the Stanford NER. To validate the effect of NER, we experiment without replacing named entities. Table 7 lists the results. The table indicates that replacing named entities improves the performance on the mail datasets. This is because names of people, locations, and organizations can be significant hints for distinguishing

Dataset	Model	ROUGE-1-F			ROUGE-2-F			ROUGE-L-F		
		# of words			# of words			# of words		
		20	40	60	20	40	60	20	40	60
ECS	IQE	<b>0.241</b>	<b>0.374</b>	<b>0.445</b>	<b>0.130</b>	<b>0.206</b>	<b>0.251</b>	<b>0.220</b>	<b>0.292</b>	<b>0.333</b>
	IQE w/o NER	0.215	0.351	0.424	0.110	0.189	0.237	0.208	0.290	0.329
	IQE w/o Pretraining	0.223	0.355	0.420	0.113	0.190	0.231	0.210	0.288	0.323
EPS	IQE	<b>0.189</b>	<b>0.292</b>	<b>0.342</b>	<b>0.091</b>	<b>0.143</b>	<b>0.168</b>	<b>0.189</b>	<b>0.268</b>	<b>0.302</b>
	IQE w/o NER	0.170	0.271	0.312	0.076	0.127	0.149	0.188	<b>0.268</b>	0.295
	IQE w/o Pretraining	0.176	0.274	0.318	0.078	0.124	0.147	0.186	0.260	0.291
TIFU	IQE	0.153	<b>0.172</b>	0.169	<b>0.024</b>	<b>0.031</b>	<b>0.033</b>	<b>0.122</b>	<b>0.136</b>	<b>0.139</b>
	IQE w/o NER	<b>0.154</b>	<b>0.172</b>	<b>0.170</b>	<b>0.024</b>	0.030	<b>0.033</b>	<b>0.122</b>	<b>0.136</b>	<b>0.139</b>
	IQE w/o Pretraining	0.143	0.161	0.160	0.020	0.027	0.029	0.116	0.131	0.133

Table 7: Results of ablation tests

correct replies. For example, if a post and a reply candidate refer to the same person’s name, the model extracts sentences that contain the person’s name. The replacement of named entities encourages the model to extract sentences semantically relevant to replies rather than simply extracting sentences that include named entities.

However, on the Reddit TIFU dataset, NER did not affect the accuracy. Reddit is an anonymized social media platform, and the posts are less likely to refer to people’s names. Thus, named entities will not be hints to predict reply-relation.

**Effect of pretraining Predictor** As explained in the section 4.3, we pretrained the Predictor in the first few epochs so that the model can learn the extraction and the prediction separately. Table 7 shows the effect of pretraining. Without pretraining, the accuracy decreased. This shows the importance of the separate training of each component.

### 5.3 Difference from Conventional Methods

As explained in the Introduction, most conventional unsupervised summarization methods are based on the assumption that important topics appear frequently in a document. TextRank is a typical example; TextRank is a centrality-based method that extracts sentences with high PageRank as the summary. A sentence having high PageRank indicates that the sentence has high similarity with many other sentences, meaning that many sentences refer to the same topic. We suspected that important topics are not always referred to frequently, and suggested another criterion: the frequency of being referred to in replies.

Comparing with TextRank, we verify that our method can capture salient sentences that the centrality-based method fails to. Figure 3 shows the correlation between the maximum PageRank in each post of ECS/EPS and ROUGE-1-F scores

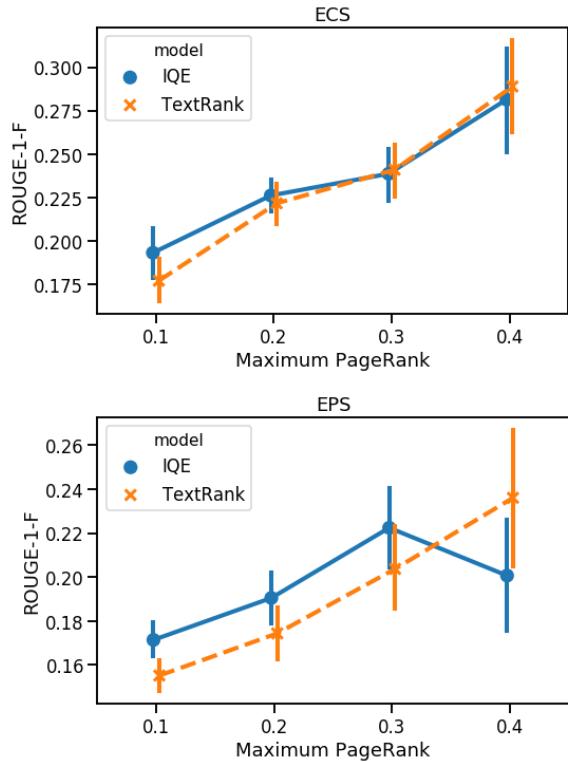


Figure 3: Correlation between ROUGE-1-F score and maximum PageRank of each post on ECS and EPS datasets. X-axis shows rounded maximum PageRank, and Y-axis shows ROUGE-1-F and the error bar represents the standard error.

of IQE and TextRank. As shown in the Figure, the ROUGE-1-F scores of our model are higher than those of TextRank when the maximum PageRank in the sentence-similarity graph is low. This supports our hypothesis that our model can capture salient sentences even when the important topic is referred to only few times.

Table 8 shows a demonstrative example of extracted summaries of IQE and TextRank. The sample is from the EPS dataset. The summary includes descriptions regarding a promotion and that the sender is having a baby. However, those words

Source Text
<i>Just got your email address from Rachel.</i>
<b>Congrats on your promotion.</b>
<i>I'm sure it's going to be a lot different for you but it sounds like a great deal.</i>
My hubby and' I moved out to Katy a few months ago. I love it there - my parents live about 10 minutes away. <b>New news from me - I'm having a baby - due in June.</b> I can't even believe it myself.
The thought of me being a mother is downright scary but I figure since I'm almost 30, I probably need to start growing up. I'm really excited though.
Rachel is coming to visit me in a couple of weeks. <b>You planning on coming in for any of the rodeo stuff?</b> <i>You'll never guess who I got in touch with about a month ago.</i>
It was the weirdest thing - heather evans. I hadn't talked to her in about 10 years. Seems like she's doing well but I can never really tell with her.
Anyway, I'll let you go. Got ta get back to work. Looking forward to hearing back from ya.
Summary (Gold)
The sender wants to congratulate the recipient for his/her new promotion, as well as, updating him/her about her life. The sender just move out to Katy few months ago. She is having a baby due in June. She is scared of being a mother but also pretty excited about it. Rachel is coming to visit her in couple of weeks and she is asking if he/she will join for any of the rodeo stuff. She run into heather evans which she hadn't talked in 10 years.

Table 8: Example of sentences extracted by **Implicit Quote Extractor (IQE) (**bold**)** and *TextRank* (*italic*).

appear only once in the source text; thus TextRank fails to capture the salient sentences. Our model, by contrast, can capture them because they are topics that replies often refer to.

## 6 Conclusion

This paper proposes Implicit Quote Extractor, a model that extracts implicit quotes as summaries. We evaluated our model with two mail datasets, ECS and EPS, and one social media dataset TIFU, using ROUGE as an evaluation metric, and validated that our model is useful for summarization. We hypothesized that our model is more likely to extract quotes and that ability improved the performance of our model. We verified these hypotheses with the Reddit TIFU dataset, but not with the email datasets, because few emails included annotated summaries, and those emails did not have replies with quotes. For future work, we will examine whether our hypotheses are valid for emails and other datasets.

## References

- Reinald Kim Amplayao and Mirella Lapata. 2020. **Unsupervised opinion summarization with noising and denoising**. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1934–1945, Online. Association for Computational Linguistics.
- Christos Baziots, Ion Androutsopoulos, Ioannis Konstas, and Alexandros Potamianos. 2019. **SEQ^3: Differentiable sequence-to-sequence-to-sequence autoencoder for unsupervised abstractive sentence compression**. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 673–681. Association for Computational Linguistics.
- Sumit Bhatia, Prakhar Biyani, and Prasenjit Mitra. 2014. **Summarizing online forum discussions – can dialog acts of individual messages help?** In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2127–2131. Association for Computational Linguistics.
- Giuseppe Carenini, Raymond T. Ng, and Xiaodong Zhou. 2007. **Summarizing email conversations with clue words**. In *Proceedings of the 16th International Conference on World Wide Web, WWW '07*, pages 91–100. ACM.
- Eric Chu and Peter J. Liu. 2019. **Meansum: A neural model for unsupervised multi-document abstractive summarization**. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pages 1223–1232.
- Günes Erkan and Dragomir R. Radev. 2004. **Lexrank: Graph-based lexical centrality as salience in text summarization**. *J. Artif. Int. Res.*, 22(1):457–479.
- Thibault Fevry and Jason Phang. 2018. **Unsupervised sentence compression using denoising auto-encoders**. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 413–422. Association for Computational Linguistics.
- Katja Filippova. 2010. **Multi-sentence compression: Finding shortest paths in word graphs**. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 322–330, Beijing, China. Coling 2010 Organizing Committee.
- Kavita Ganeshan. 2015. **Rouge 2.0: Updated and improved measures for evaluation of summarization tasks**.
- Demian Gholipour Ghalandari. 2017. **Revisiting the centroid-based method: A strong baseline for multi-document summarization**. In *Proceedings of the Workshop on New Frontiers in Summarization*,

- pages 85–90. Association for Computational Linguistics.
- Aria Haghighi and Lucy Vanderwende. 2009. **Exploring content models for multi-document summarization**. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 362–370, Boulder, Colorado. Association for Computational Linguistics.
- Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. 2017. **A joint many-task model: Growing a neural network for multiple NLP tasks**. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1923–1933, Copenhagen, Denmark. Association for Computational Linguistics.
- Zhanying He, Chun Chen, Jiajun Bu, Can Wang, Li-jun Zhang, Deng Cai, and Xiaofei He. 2012. **Document summarization based on data reconstruction**. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, AAAI’12, pages 620–626. AAAI Press.
- Masaru Isonuma, Junichiro Mori, and Ichiro Sakata. 2019. **Unsupervised neural single-document summarization of reviews via learning latent discourse structure and its ranking**. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2142–2152. Association for Computational Linguistics.
- Eric Jang, Shixiang Gu, and Ben Poole. 2017. **Categorical reparameterization with gumbel-softmax**. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.
- Mikael Kågebäck, Olof Mogren, Nina Tahmasebi, and Devdatt Dubhashi. 2014. **Extractive summarization using continuous vector space models**. In *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)*, pages 31–39. Association for Computational Linguistics.
- Byeongchang Kim, Hyunwoo Kim, and Gunhee Kim. 2019. **Abstractive summarization of Reddit posts with multi-level memory networks**. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2519–2531. Association for Computational Linguistics.
- Diederik P. Kingma and Jimmy Ba. 2015. **Adam: A method for stochastic optimization**. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.
- Philippe Laban, Andrew Hsi, John Canny, and Marti A. Hearst. 2020. **The summary loop: Learning to write abstractive summaries without examples**. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5135–5150, Online. Association for Computational Linguistics.
- Chin-Yew Lin. 2004. **ROUGE: A package for automatic evaluation of summaries**. In *Text Summarization Branches Out*, pages 74–81. Association for Computational Linguistics.
- He Liu, Hongliang Yu, and Zhi-Hong Deng. 2015. **Multi-document summarization based on two-level sparse representation model**. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI’15, pages 196–202. AAAI Press.
- Vanessa Loza, Shibamouli Lahiri, Rada Mihalcea, and Po-Hsiang Lai. 2014. **Building a dataset for summarization and keyword extraction from emails**. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*, pages 2441–2446. European Languages Resources Association (ELRA).
- Shulei Ma, Zhi-Hong Deng, and Yunlun Yang. 2016. **An unsupervised multi-document summarization framework based on neural document model**. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1514–1523. The COLING 2016 Organizing Committee.
- Yashar Mehdad, Giuseppe Carenini, and Raymond T. Ng. 2014. **Abstractive summarization of spoken and written conversations based on phrasal queries**. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1220–1230. Association for Computational Linguistics.
- Rada Mihalcea and Paul Tarau. 2004. **TextRank: Bringing order into text**. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411. Association for Computational Linguistics.
- Tatsuro Oya and Giuseppe Carenini. 2014. **Extractive summarization and dialogue act modeling on email threads: An integrated probabilistic approach**. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 133–140. Association for Computational Linguistics.
- Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. **A decomposable attention model for natural language inference**. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2249–2255. Association for Computational Linguistics.
- Guokan Shang, Wensi Ding, Zekun Zhang, Antoine Tixier, Polykarpos Meladianos, Michalis Vazirgiannis, and Jean-Pierre Lorré. 2018. **Unsupervised abstractive meeting summarization with multi-sentence compression and budgeted submodular**

**maximization.** In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 664–674. Association for Computational Linguistics.

Wenpeng Yin and Yulong Pei. 2015. **Optimizing sentence modeling and selection for document summarization.** In *Proceedings of the 24th International Conference on Artificial Intelligence, IJCAI'15*, pages 1383–1389. AAAI Press.

Hao Zheng and Mirella Lapata. 2019. **Sentence centrality revisited for unsupervised summarization.** In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6236–6247, Florence, Italy. Association for Computational Linguistics.

## A Appendices

### A.1 Decomposable Attention

As explained in section 3, the Predictor uses Decomposable Attention for prediction. Decomposable Attention computes a two-dimensional attention matrix, computed by two sets of vectors, and thus, captures detailed information useful for prediction. The computation uses the following equations:

$$E_{tj} = (\mathbf{x}_t^{ext})^T \mathbf{h}_j^r \quad (13)$$

$$\beta_t = \sum_{j=1}^M \frac{\exp(E_{tj})}{\sum_{k=1}^M \exp(E_{tk})} \mathbf{h}_j^r \quad (14)$$

$$\alpha_j = \sum_{t=1}^L \frac{\exp(E_{tj})}{\sum_{k=1}^L \exp(E_{kj})} \mathbf{x}_t^{ext} \quad (15)$$

The computation of  $\mathbf{x}_t^{ext}$  and  $\mathbf{h}_j^r$  are explained in section 3. First, we compute a co-attention matrix  $E$  as in (13). The weights of the co-attention matrix are normalized row-wise and column-wise in the equations (14) and (15).  $\beta_i$  is a linear sum of reply features  $\mathbf{h}_j^r$  that is aligned to  $\mathbf{x}_t^{ext}$  and vice versa for  $\alpha_j$ .

$$\mathbf{v}_{1,t} = G([\mathbf{x}_t^{ext}; \beta_t]) \quad \mathbf{v}_{2,j} = G([\mathbf{h}_j^r; \alpha_j]) \quad (16)$$

$$\mathbf{v}_1 = \sum_{t=1}^L \mathbf{v}_{1,t} \quad \mathbf{v}_2 = \sum_{j=1}^M \mathbf{v}_{2,j} \quad (17)$$

$$y = \text{sigmoid}(H([\mathbf{v}_1; \mathbf{v}_2])) \quad (18)$$

Next, we separately compare the aligned phrases  $\beta_t$  and  $\mathbf{x}_t^{ext}$ ,  $\alpha_j$  and  $\mathbf{h}_j^r$ , using a function  $G$ .  $G$  denotes a feed-forward neural network, and  $[;]$  denotes concatenation. Finally, we concatenate  $\mathbf{v}_1$  and  $\mathbf{v}_2$  and obtain binary-classification result  $y$  through a linear layer  $H$  and the sigmoid function.

# Unsupervised Aspect-Level Sentiment Controllable Style Transfer

Mukuntha N S<sup>§</sup>, Zishan Ahmad<sup>§</sup>, Asif Ekbal, Pushpak Bhattacharyya

Department of Computer Science and Engineering

Indian Institute of Technology Patna

Bihar, India

{mukuntha.cs16, 1821cs18, asif, pb}@iitp.ac.in

## Abstract

Unsupervised style transfer in text has previously been explored through the sentiment transfer task. The task entails inverting the overall sentiment polarity in a given input sentence, while preserving its content. From the Aspect-Based Sentiment Analysis (ABSA) task, we know that multiple sentiment polarities can often be present together in a sentence with multiple aspects. In this paper, the task of aspect-level sentiment controllable style transfer is introduced, where each of the aspect-level sentiments can individually be controlled at the output. To achieve this goal, a BERT-based encoder-decoder architecture with saliency weighted polarity injection is proposed, with unsupervised training strategies, such as ABSA masked-language-modelling. Through both automatic and manual evaluation, we show that the system is successful in controlling aspect-level sentiments.

## 1 Introduction

With a rapid increase in the quality of generated text, due to the rise of neural text generation models (Kalchbrenner and Blunsom, 2013; Cho et al., 2014; Sutskever et al., 2014; Vaswani et al., 2017), controllable text generation is quickly becoming the next frontier in the field of text generation. Controllable text generation is the task of generating realistic sentences whose attributes can be controlled. The attributes to control can be: (i). *Stylistic*: Like politeness, sentiment, formality etc, (ii). *Content*: Like information, entities, keywords etc. or (iii). *Ordering*: Like ordering of information, events, plots etc.

Controlling sentence level polarity has been well explored as a style transfer task. Zhang et al. (2018) used unsupervised machine translation techniques for polarity transfer in sentences. Yang et al. (2018)

<sup>§</sup>equal contribution

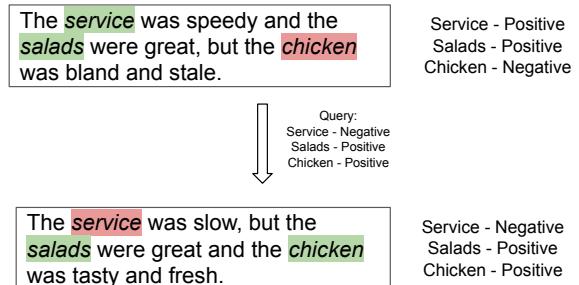


Figure 1: An example of the proposed aspect-level sentiment style transfer task

used language models as discriminators to achieve style (polarity) transfer in sentences. Li et al. (2018a) proposed a simpler method where they deleted the attribute markers and devise a method to replace or generate the target attribute-key phrases in the sentence.

In this paper we explore a more fine-grained style transfer task, where each aspect's polarities can be changed individually. Recent interest in Aspect-Based Sentiment Analysis (ABSA) (Pontiki et al., 2014) has shown that sentiment information can vary within a sentence, with differing sentiments expressed towards different aspect terms of target entities (e.g. ‘food’, ‘service’ in a restaurant domain). We introduce the task of aspect-level sentiment transfer - the task of rewriting sentences to transfer them from a given set of aspect-term polarities (such as ‘positive sentiment’ towards the service of a restaurant and a ‘positive sentiment’ towards the taste of the food) to a different set of aspect-term polarities (such as ‘negative sentiment’ towards the service of a restaurant and a ‘positive’ sentiment towards the taste of the food). This is a more challenging task than regular style transfer as the style attributes here are not the overall attributes for the whole sentence, but are localized to specific parts of the sentence, and multiple opposing at-

tributes could be present within the same sentence. The target of the transformation made needs to be localized and the other content expressed in the rest of the sentence need to be preserved at the output. An example of the task is shown in Figure 1.

For successful manipulation of the generated sentences, a few challenges need to be addressed: (i). The model should learn to associate the right polarities with the right aspects. (ii). The model needs to be able to correctly process the aspect-polarity query and accordingly delete, replace and generate text sequence to satisfy the query. (iii). The polarities of the aspects not in the query should not be affected. (iv). The non-attribute content and fluency of the text should be preserved.

We explore this task in an unsupervised setting (as is common with most style-transfer tasks due to the lack of an aligned parallel corpus) using only monolingual unaligned corpora. In this work, a novel encoder-decoder architecture is proposed to perform unsupervised aspect-level sentiment transfer. A BERT (Devlin et al., 2019) based encoder is used that is trained to understand aspect-specific polarity information. We also propose using a ‘polarity injection’ method, where saliency-weighted aspect-specific polarity information is added to the hidden representations from the encoder to complete the query for the decoder.

## 1.1 Motivation

The Aspect-Based Sentiment Analysis (ABSA) task shows that differing sentiments can be present within the same sentence, localized to different entities or parts of the text. The notion of styles in natural language can be used to refer to the attributes, such as sentiment, formality in content, emotion, sarcasm, etc. Similar to the sentiment, these other attributes can also be present localized to different entities taking differing values at each location. If we consider the style ‘emotion’ with the example “Although Alice infuriates me with her prattle and Bob scares me, I am quite happy about how things are turning out.” - A single piece of text (such as a single sentence) can express an emotion, such as ‘happiness’ about an event while expressing ‘fear’ towards some entity and ‘anger’ towards a second entity. This shows that style transfer in language needs a more nuanced understanding. Especially when generating larger pieces of text, multiple such styles could intermingle, and differing styles can often be present together when discussing different

topics and entities. Our work intends to take the first step towards a more controllable form of fine-grained style transfer with the task of aspect-level sentiment style transfer.

## 2 Related Work

In this section we present an overview of the related literature.

### 2.1 Sentiment Transfer

To the best of our knowledge, our current work is the first to tackle aspect-level sentiment transfer. Most of the previous works involving sentiment transfer (Li et al., 2018b; Yang et al., 2018; Shen et al., 2017; Xu et al., 2018; Prabhumoye et al., 2018; Wu et al., 2019) consider the style that is present throughout the sentence and seek to transfer only the overall sentiment polarities expressed. Tian et al. (2018) proposed a new training objective for content preservation during style transfer. They used Part-of-Speech (PoS) tagging to collect nouns at inputs, and expect them to be present at the output for content preservation. To achieve this, they proposed a PoS preservation constraint and ‘Content Conditional Language Modelling’. They tested their system on sentiment style transfer task.

Wang et al. (2019) proposed a method that can also control the degree of polarity transfer in a sentence with multiple aspect categories present in it. Unlike their task which deals with *predefined aspect categories*, our task deals with *opinion target expressions*. Aspect categories are coarse entities that are few in number and predefined for a certain domain, while aspect-terms or opinion target expressions are fine-grained entities that are present in the text. They also did not investigate selectively transferring the polarity over a subset of aspects with multiple differing polarities at the output and only invert the overall polarity expressed by the sentence. Our method works across thousands of unique opinion target expressions (Table 1 shows the number unique target aspects present in each of our datasets). Our method also does not need these to be predefined, and so could be used to control the polarities of previously unseen target expressions as well.

### 2.2 Unsupervised Machine Translation

Previous works in unsupervised neural machine translation (Artetxe et al., 2017) and unsupervised style transfer (Zhang et al., 2018) have shown that,

with only monolingual data, using a denoising auto-encoder loss and an on-the-fly back-translation loss can be very successful in achieving transfer. Both of these training steps are used as part of our method to train the network in an unsupervised fashion.

### 2.3 Natural Language Generation Architecture

Lai et al. (2019) proposed an adversarial training mechanism for Gated Recurrent Unit (GRU) based encoder-decoder model for sentiment polarity transfer and multiple-attribute transfer tasks. They split the training mechanism of their model into two phases, *viz.* (i). Style transfer phase and (ii). Reconstruction phase. Pryzant et al. (2020) proposed a method to remove subjective bias in the sentences. They proposed adding a ‘join-embedding’ weighted by a word subjective-bias probability to automatically edit the hidden states from the encoder. We adapt this ‘join-embedding’ method to inject weighted polarities into our encoder outputs as described in Section 3.5.

### 2.4 Aspect Based Sentiment Analysis

Aspect based sentiment analysis (ABSA) has been explored in a series of SemEval shared tasks. The task consists of both aspect term extraction and aspect sentiment prediction. Tay et al. (2018) proposed ‘Aspect Fusion LSTM’ to attend on the associative relationships between sentence words and aspect words to classify aspect polarities. Xu et al. (2019) proposed BERT based models for aspect term extraction and aspect-polarity classification tasks. We build similar BERT based aspect term-extraction and aspect-polarity classification models and use them to label Yelp reviews dataset. This dataset is then used for aspect-level sentiment controllable style transfer task in this paper.

## 3 Methodology

### 3.1 Problem Statement

Let us assume we have access to a corpora of labelled sentences  $D = (x_1, l_1) \dots (x_n, l_n)$ , where  $x_i$  is a sentence, and  $l_i = \{(t_{i1}, p_{i1}) \dots (t_{im}, p_{im})\}$ . Here,  $t_{ij}$  is an aspect-target or ‘Opinion Target Expression’ (Pontiki et al., 2014), and  $p_{ij}$  is the corresponding sentiment-polarity expressed towards  $t_{ij}$ , where  $p_{ij} \in \{“positive”, “negative”\}$ . A model is to be learned that takes as input  $(x, l_{tgt})$  where  $x$

is the source sentence expressing some aspect-polarity set  $l_{src}$ , and outputs  $y$  that retain all the non-polarity content in  $x$  while expressing the aspect-polarity set  $l_{tgt}$ .

This is to be performed in an unsupervised manner, where we do not assume access to an aligned set of parallel sentences with the same content but different aspect-polarities.

The overall architecture consists of a Transformer (Vaswani et al., 2017) encoder-decoder neural network, where the encoder is BERT (Devlin et al., 2019). In this section, we describe the architecture and the training methodology used. The inputs provided to the model are the sentence, a list of aspects, their corresponding desired target polarities  $l_{tgt}$  and their corresponding per-token weights (explained in Section 3.5).

### 3.2 ABSA Input Representation

The BERT model (Devlin et al., 2019) was originally trained with two objectives: (i). A cloze objective where the classifier predicts missing words in a sequence, and (ii). A sentence-pair classification objective. For the sentence-pair objective, BERT was trained to take inputs as segment-pairs, where each segment has a different embedding added to it and are separated by a [SEP] token. For our input representation, we construct such segment-pairs. The first segment consists of an aspect-polarity sequence  $SEG_A = “T_1P_1[SEP_{ASP}] \dots T_kP_k[SEP_{ASP}]”$ , where  $T_i$  is the tokenized target aspect term and  $P_i \in \{[POS], [NEG]\}$  is a polarity corresponding to it.  $[SEP_{ASP}]$  is a separator token.  $[POS]$ ,  $[NEG]$  are the special tokens corresponding to the ‘positive’ and ‘negative’ sentiments, for which the unused tokens from the BERT vocabulary were used. The second segment  $SEG_B$  consists of a sentence expressing some sentiment towards these targets.

### 3.3 Preconditioning the BERT-encoder for ABSA Input Representations

We precondition the BERT encoder to better understand the ABSA task and to learn the token-embeddings for  $[POS]$  and  $[NEG]$  with MLM pre-training (a cloze objective) (c.f. Figure 2). For each data instance, with an equal probability, we randomly mask out either (i). all the polarity tokens from aspect-polarity sequence ( $SEG_A$ ), or (ii). random tokens from the sentence ( $SEG_B$ ) and train the encoder to correctly predict the masked-

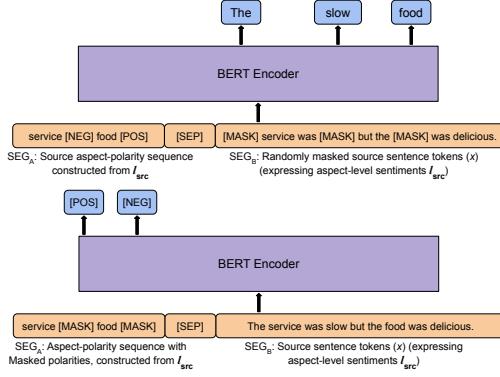


Figure 2: BERT Encoder pre-training

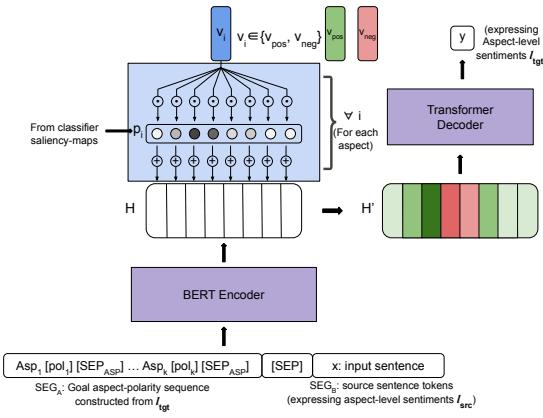


Figure 3: The encoder-decoder network used, with the polarity injection.

out tokens. When the polarities get masked, the encoder learns to correctly understand the aspect-level sentiment polarities from a sentence. When the words from the sentence get masked, the encoder also learns to correctly predict attribute markers corresponding to a given aspect and a sentiment. For example, it would learn associations between the markers, such as ‘personable’ (or ‘rude’) when given an aspect-term, such as ‘staff’ with a polarity ‘[POS]’ (or ‘[NEG]’) as opposed to an aspect-marker, such as ‘delicious’, which cannot be used with ‘staff’.

### 3.4 Encoder-Decoder Architecture

To convert a sentence from one set of aspect-level polarities to another, the input to the encoder consists of the target aspect-level polarities  $l_{tgt}$  as  $SEG_A$  with the source sentence  $x$  passed as  $SEG_B$ . The full architecture is shown in Figure 3. The source sentence  $x$  expresses some source aspect-level polarities  $l_{src}$  which is not provided to the model. The polarity-injection (explained in Section 3.5) adds the weighted target polarities  $l_{tgt}$

into the hidden-representation  $H$  from the BERT-encoder to obtain  $H'$  which is passed to the decoder. The decoder is trained to output the target sentence  $y$  which consists of the same content as present in  $x$  but expressing the target aspect-level polarities  $l_{tgt}$ . This architecture is trained in an unsupervised fashion as explained in Section 3.6.

### 3.5 Polarity Injection

In Pryant et al. (2020), authors showed that the hidden states of the encoder can be edited by adding weighted vectors to indicate subjective-bias, before being input to the decoder. They proposed this as a method to join the results from two sub-modules in their system. Here, we extend this to cover multiple attributes - the ‘positive’ and ‘negative’ sentiments, and substitute the supervised model they train with saliency-based weights. We inject (add) weighted amounts of two vectors corresponding to these two attributes to edit the hidden states output by the encoder. For each aspect, the vector added corresponds to the desired target polarity of this aspect, and the amount added to a given token depends on the saliency-based weight for this token calculated from the gradient for this aspect’s polarity from a classification model (described in Section 4.1.1).

More formally, the polarity injection is calculated from equation 1.  $H = [h_1, h_2, \dots, h_k]$  that denotes the hidden-state output from the encoder, and  $H' = [h'_1, h'_2, \dots, h'_k]$  are the new hidden-states calculated after polarity-injection. The number  $p_{ij}$  denotes the saliency-based weightage for token  $j$  with respect to aspect  $i$ . Figure 3 shows the polarity-injection architecture.  $v_{pos}$  and  $v_{neg}$  are the special vector-embeddings, which have the same size as the hidden dimension, and trained to denote the positive and negative sentiment, respectively.

$$h'_j = h_j + \sum_{i=1}^k p_{ij} \cdot v_i \quad (1)$$

$$v_i = \begin{cases} v_{pos} & \text{if } pol_i \text{ desired is positive} \\ v_{neg} & \text{if } pol_i \text{ desired is negative} \end{cases} \quad (2)$$

where  $pol_i$  is the target (desired) polarity from  $l_{tgt}$  for the  $i^{th}$  aspect-term. For calculating  $p_{ij}$ , saliency-maps obtained for each aspect from the polarity classifier described in 4.1 are used. Saliency-maps (Simonyan et al., 2014) are calculated with the gradient of the loss at the input, as given in equation

Dataset	No. of Sentences	No. of Target Aspects	No. of Unique Target Aspects
SemEval (Train and Validation)	2,242	4,016	1,437
SemEval (Test)	401	513	269
Yelp (Train and Validation)	361,968	471,820	47,750

Table 1: Data distribution for the restaurant domain. The Yelp dataset does not contain target aspects and their polarities extracted, and these were extracted with a classifier trained on the SemEval training data

3. The  $s_{tok}$  values for all the tokens  $tok$  in the sentence  $x$  are normalized between 0 and 1 for each (target  $t$ , sentence  $x$ ) pair to obtain the  $p_{ij}$  values. Since the saliency-maps produce high values for the tokens that are important in calculating the sentiment’s polarity, adding the ‘positive’ or ‘negative’ embedding weighted by these probabilities would provide hints to the decoder about the important words to be rewritten with the required sentiment. The  $p_{ij}$  values over Segment-A is set to 1 over all the tokens corresponding to the  $i^{\text{th}}$  aspect term ( $T_i P_i$ ) (see Section 3.2) and 0 over the other tokens.

$$s_{tok} = \left| \frac{\partial L(y_t; x, t, \theta)}{\partial emb_{tok}} \right|; \forall tok \in x \quad (3)$$

**One-Zero Alternative to Saliency:** To test for performance in the absence of any saliency information, we also propose using a one-zero setup. Here,  $p_{ij}$  is set to 1 over the tokens corresponding to the  $i^{\text{th}}$  aspect-term and 0 elsewhere. So in this setup,  $v_{pos}$  gets added to the tokens corresponding to the positive aspects and  $v_{neg}$  gets added to the tokens corresponding to the negative aspects. For example, in Figure 1,  $v_{pos}$  gets added to the sub-word tokens corresponding to the word ‘salads’ and ‘chicken’, while  $v_{neg}$  gets added to the sub-word tokens corresponding to the word ‘service’.

### 3.6 Unsupervised Training

For training the model in an unsupervised setting, we alternate training steps between a denoising auto-encoding objective and a back-translation objective. During the denoising step, we add random noise to the sentence part of the input  $SEG_B$ . We also randomly mask the polarities in the aspect-polarity sequence in  $SEG_A$  with a small probability to ensure the model learns to generate outputs using the polarity injection clues. During the back-translation step, a random query  $l_{tgt}$  aspect-polarity sequence is used to produce an intermediate translation (using the model), and the same model is trained to regenerate the original input when provided the aspect-polarity sequence from the original input sentence.

## 4 Experiments

In this section we report the datasets used for the experiments and the implementation details.

### 4.1 Datasets

Text generation tasks require huge amounts of data, however there are no aspect-sentiment annotated datasets that are large enough for our task. Fortunately, aspect extraction and aspect-sentiment classification tasks have been well explored and have several publicly available datasets. We used datasets (only restaurant domain) from SemEval 2014, 2015 and 2016 (ABSA task) to train BERT based aspect extraction and aspect-sentiment classification systems. We only consider positive and negative polarities for our experiments.

For the task of aspect-level sentiment style transfer, we use Yelp dataset. Since this dataset does not contain aspect-level polarity information or the target-aspects extracted, we use our BERT-based target-extraction model and BERT-based polarity classification model which were trained on the SemEval ABSA training data, to generate aspect-level sentiment data from the Yelp reviews dataset. Table 1 shows some statistics from the datasets.

#### 4.1.1 Aspect based Sentiment Analysis with BERT

A pipeline of BERT-based models was trained for target-extraction and aspect-level polarity classification over the SemEval dataset. These are the models used to extract target-aspects and their polarities from the Yelp dataset. The target extraction task was posed as a sequential token classification problem with BERT using the IOB2 format (SANG, 1999). This BERT model was fed the whole sentence as the input segment and it obtained an F1-score of 0.8012 (evaluation carried out similar to Sang and Buchholz (2000)). The sentiment-polarity prediction task is posed as a sentence-pair classification problem using BERT, with the sentence provided as the first segment and the aspect-term as the second segment. This model obtained an F1-score of 0.9080 for the positive po-

Model	Classifier Score (Overall)	Classifier Score (1-Aspect)	Classifier Score (2-Aspects)	Classifier Score (3-or-more Aspects)	BLEU Score
BERT-Baseline (BB)	0.5158	0.4983	0.5448	0.5036	36.0683
BB + MLM pretraining (BB-MLM)	0.5298	0.5433	0.5310	0.5145	35.4601
BB-MLM + one-zero polarity injection	0.5415	0.5675	0.5276	0.5290	35.8244
BB-MLM + saliency-based polarity injection	<b>0.5918</b>	<b>0.6125</b>	<b>0.5828</b>	<b>0.5797</b>	<b>39.3838</b>

Table 2: Results of automatic evaluation. The overall classifier score is calculated over all queries. The other columns show the score calculated only on queries with one aspect, two aspects or three or more aspects. The classifier scores are calculated on the full test set, while the BLEU scores are measured with reference-outputs for a subset of 100 queries.

Model	Att	Con	Gra
BERT-Baseline (BB)	2.48	3.99	3.96
BB + MLM pretraining (BB-MLM)	2.64	3.95	4.04
BB-MLM + one-zero polarity injection	2.80	4.00	<b>4.05</b>
BB-MLM + saliency-based polarity injection	<b>2.98</b>	<b>4.08</b>	<b>4.05</b>

Table 3: Results of manual evaluation. Here, ‘Att’ stands for attribute match, ‘Con’ stands for content preservation and ‘Gra’ stands for grammaticality or fluency. Manual evaluation is performed on a subset of 100 queries from all the test set queries, and averaged scores are shown.

larity and 0.8239 for the negative polarity on the ABSA restaurant dataset. Using this classifier, for each (Sentence, Target) pair the gradient of the loss was taken at the input token embeddings and normalized to obtain the saliency-based weights used for polarity-injection.

## 4.2 Implementation Details

All the models were implemented using PyTorch (Paszke et al., 2017). The BERT model was implemented using the transformers library (Wolf et al., 2019). Models are trained with an initial learning rate of 1e-4 with a linear schedule and a warmup (Vaswani et al., 2017), using the Adam Optimizer (Kingma and Ba, 2019). Mini-batches of size 32 were used during training. A linear schedule was used for the weight of the loss from the denoising auto-encoding step, which was set to decrease from 1 to 0.1 for the first 30,000 optimization steps and then decrease linearly to 0 over the next 70,000 steps. The models were each trained for 8 epochs on the Yelp dataset. The random masking probability used during pre-training was 0.25. During the denoising step, a probability of 0.25 was used for dropping words, and words were shuffled with a window-size of 3.

## 5 Results and Analysis

### 5.1 Evaluation

The evaluation metrics we use are an extension of the metrics used for evaluating the sentiment transfer task by previous work (such as Li et al. (2018b); Wang et al. (2019)). The evaluation was done with the SemEval test dataset. Queries were generated from this data by randomly inverting a subset (non-null, improper subset) of the polarities expressed at the input. For queries with 2 or more aspects, as many queries were generated as there were aspects in the sentence with different random inversions, resulting in a total of 513 evaluation queries. A sample consisting of 100 queries from the test set was used for manual evaluation.

#### 5.1.1 Automatic Evaluation

For automatic evaluation, we use a classifier score and a BLEU score. The results for automatic evaluation are shown in Table 2.

**Classifier Score:** We use an aspect-level sentiment polarity classifier to measure how many of the outputs express the necessary target polarities (Li et al., 2018b). We use the classifier described in 4.1.1 for the polarity prediction. We define the *classifier score* to be the fraction of aspect-level sentiment polarities (predicted by the classifier from the output) that match with the desired aspect-level polarity (from the query). While averaging, each

Input	Query	Model Output
overall, decent food at a good price, with friendly people.	food - negative people - positive	overall , mediocre food at a good price , with friendly people .
	food - positive people - negative	overall , decent food at a good price , with rude people .
the waiter was attentive , the food was delicious and the views of the city were great	waiter - negative food - positive views of the city - positive	the waiter was inattentive , the food was delicious and the views of the city were great .
	waiter - positive food - negative views of the city - positive	the waiter was attentive , the food was disappointing but the views of the city were great .
	waiter - positive food - negative views of the city - negative	the waiter was attentive , the food was disappointing but the views of the city were terrible .

Table 4: Example outputs from the full model with saliency-based polarity injection with different aspect-level polarity queries.

aspect-level sentiment in a query was treated as a separate instance.

**BLEU Scores:** Like in Li et al. (2018b); Gan et al. (2017), human reference outputs were written for 100 of the queries. Three Human experts were asked to rewrite the reviews with as much content preserved as possible, without compromising fluency. These experts had good language abilities and having satisfactory knowledge in the relevant area. We report BLEU scores for the models against these references. A BLEU score could be treated as a measure of content preservation from the input or the output fluency.

### 5.1.2 Manual Evaluation

Following the previous methods (Li et al., 2018b; Wang et al., 2019) for manual evaluation of style transfer, workers were asked to rate the output sentences on the Likert-scale (1 to 5) for three criteria - Attribute match to the query set of aspect-level polarities (Att), Fluency (Gra) measuring the naturalness of the output and Content preservation (Con). They were shown the source sentence, the query aspect-level polarities and the model output. The results of manual evaluation are shown in Table 3<sup>1</sup>

### 5.2 Error Analysis

The importance of each component in our model is shown through an ablation study in Table 2 and Table 3. From the classifier-based score, we see that the full model with saliency-based polarity injection is the most successful in transferring sentiment-level polarities. Polarity injection, even without

<sup>1</sup>Inter-annotator agreement measured through the Krippendorff’s alpha was found to be 0.92, 0.82, 0.87 for ‘Att’, ‘Con’, and ‘Gra’ respectively.

saliency information is seen to be useful. The models with polarity injection are especially better at transferring sentiments when three or more aspects are present, showing that the polarity signals are useful in localizing the style attributes with multiple targets present. The model using saliency-based weighting for the polarity injection has a significantly higher classifier score. This could be because of the saliency information acting like an adversarial white-box attack on the classifier, making it easier to obtain higher classifier scores.

The Content preservation (Con) scores and BLEU scores for the baseline models are significantly high, but these models also show poor Attribute match (Att) scores. This means that many of the sentiments at the output were left untransferred resulting in the poor Att score, while large parts of the input text were copied over to the output resulting in the larger BLEU and Content preservation scores. The improved Content preservation (Con) scores and the Fluency (Gra) from the model without saliency information to the model with saliency-based weighting shows that the attribute transfer with saliency-based info is more successful in inverting the correct polarities, while maintaining the content and fluency, due to the added information about the words to be edited.

The Table 5 shows how the model outputs change with different components of the model are ablated. With queries involving mostly positive or negative attributes, the saliency-based polarity injection supports the localized inversion of sentiment in the output. Outputs also show how polarity injection helps produce the required change with more content and fluency preserved, by selectively editing the correct words.

<i>Input</i>	the <b>veal</b> and the <b>mushrooms</b> were cooked perfectly .
<i>Query</i>	<b>veal</b> - positive, <b>mushrooms</b> - negative
<b>BERT-Baseline (BB)</b>	the <b>veal</b> and the <b>mushrooms</b> were not cooked perfectly .
<b>BB + MLM pretraining (BB-MLM)</b>	the <b>veal</b> and the <b>mushrooms</b> were over cooked perfectly .
<b>BB-MLM + one-zero polarity injection</b>	the <b>veal</b> was gross and the <b>mushrooms</b> were over cooked .
<b>BB-MLM + saliency-based polarity injection</b>	loved the <b>veal</b> and the <b>mushrooms</b> were over cooked .
<i>Input</i>	the <b>waiter</b> was attentive , the <b>food</b> was delicious and the <b>views of the city</b> were great .
<i>Query</i>	<b>waiter</b> - negative, <b>food</b> - positive, <b>views of the city</b> - positive
<b>BERT-Baseline (BB)</b>	the <b>waiter</b> was attentive , the <b>food</b> was delicious and the <b>city</b> were great .
<b>BB + MLM pretraining (BB-MLM)</b>	the <b>waiter</b> was attentive , the <b>food</b> was delicious and the <b>views of the city</b> were great .
<b>BB-MLM + one-zero Polarity injection</b>	the <b>waiter</b> was attentive , the <b>food</b> was delicious and the <b>views of the city</b> were great .
<b>BB-MLM + saliency-based polarity injection</b>	the <b>waiter</b> was inattentive , the <b>food</b> was delicious and the <b>views of the city</b> were great .
<i>Input</i>	for 7 years they have put out the most tasty, most delicious <b>food</b> and kept it that way...
<i>Query</i>	<b>food</b> - negative
<b>BERT-Baseline (BB)</b>	for 7 years they have put out the most tasty <b>food</b> and kept it that way.
<b>BB + MLM pretraining (BB-MLM)</b>	for 7 years they have put out the most greasy <b>food</b> and bland <b>food</b> that way...
<b>BB-MLM + one-zero polarity injection</b>	for 6 years they have put out the most bland <b>food</b> and kept it that way...
<b>BB-MLM + saliency-based polarity injection</b>	for 7 years they have put out the most bland <b>food</b> and kept it that way...

Table 5: Example outputs from the SemEval data showing aspect-level sentiment transfer from the ablated models. Aspects colored red (negative) or green (positive) indicate their sentiment.

<i>Input</i>	i must say i am surprised by the bad reviews of the <b>restaurant</b> earlier in the year , though .
<i>Query</i>	<b>restaurant</b> - negative
<i>Output</i>	i must say i am surprised by the bad <b>reviews</b> of the restaurant earlier in the year , though .
<i>Comment</i>	No change. Sentiment here is implied and latent.
<i>Input</i>	the <b>space</b> is limited so be prepared to wait up to 45 minutes - 1 hour , but be richly rewarded when you savor the delicious <b>indo-chinese food</b>
<i>Query</i>	<b>space</b> - positive, <b>indo-chinese food</b> - positive
<i>Output</i>	the <b>space</b> is extensive so be prepared to 10 - 15 - 20 + minutes , but delicious <b>chinese food</b> .
<i>Comment</i>	Disfluency and dropped content due to the length of input and the negative sentiment implied through the word ‘waiting’.
<i>Input</i>	i'd be horrified if my <b>staff</b> were turning away customers so early and so rudely!
<i>Query</i>	<b>staff</b> - positive
<i>Output</i>	i'd be delighted if my <b>staff</b> were turning away customers so early and nicely!
<i>Comment</i>	Lower naturalness of the output from real-world knowledge that turning away customers is bad.
<i>Input</i>	i had <b>fish</b> and my husband had the <b>filet</b> - both of which exceeded our expectations .
<i>Query</i>	<b>fish</b> - negative, <b>filet</b> - positive
<i>Output</i>	i had <b>fish</b> and my husband had the <b>filet</b> - both of which exceeded our expectations .
<i>Comment</i>	The attribute markers for fish and filet are shared, making transfer difficult. Significant rewriting of the input is needed to produce acceptable fluent output.

Table 6: Example sentences that show difficulty in transferring sentiment.

To understand the errors in outputs better, the outputs marked with low Att, Con and Gra scores were examined. Some of these outputs are shown and discussed in Table 6. Many of the failures in Attribute match were found to be due to the complexity involved in the language, such as when the sentiment expressed towards a target is implicit from the content of the review. The absence of

attribute markers also makes it harder to convert sentiment. Most outputs with low Con and Gra scores were found to contain very long sentences. The models were trained on the Yelp dataset which mostly contained smaller sentences. Failed examples with multiple different polarities at the output were often also due to the attribute markers towards different aspects being shared in the input sentence.

Such examples require significant rewriting and reordering to produce sentences of acceptable fluency, and our method seems most successful when making localized changes such as with word replacements.

## 6 Conclusion and Future Work

In this paper, the task of aspect-level sentiment style transfer has been introduced, where stylistic attributes can be localized to different parts of a sentence. We have proposed a BERT-based encoder-decoder architecture with saliency-based polarity injection and show that it can be successful at the task when trained in an unsupervised setting. The experiments have been conducted on an aspect level polarity tagged benchmark dataset related to the restaurant domain. This work is hopefully an important initial step in developing a fine-grained controllable style transfer system. In the future, we would like to explore the ability to transfer such systems to data-sparse domains, and explore injecting attributes such as emotions to targets attributes in larger pieces of text.

## Acknowledgments

Authors duly acknowledge the support from the Project titled Sevak-An Intelligent Indian Language Chatbot, Sponsored by SERB, Govt. of India (IMP/2018/002072).

## References

- Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2017. Unsupervised neural machine translation. *arXiv preprint arXiv:1710.11041*.
- Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*.
- Chuang Gan, Zhe Gan, Xiaodong He, Jianfeng Gao, and Li Deng. 2017. Stylenet: Generating attractive visual captions with styles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3137–3146.
- Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–1709.
- Diederik P Kingma and J Adam Ba. 2019. A method for stochastic optimization. *arxiv* 2014. *arXiv preprint arXiv:1412.6980*, 434.
- Chih-Te Lai, Yi-Te Hong, Hong-You Chen, Chi-Jen Lu, and Shou-De Lin. 2019. Multiple text style transfer by using word-level conditional generative adversarial network with two-phase training. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3570–3575.
- Juncen Li, Robin Jia, He He, and Percy Liang. 2018a. Delete, retrieve, generate: A simple approach to sentiment and style transfer. *arXiv preprint arXiv:1804.06437*.
- Juncen Li, Robin Jia, He He, and Percy Liang. 2018b. Delete, retrieve, generate: A simple approach to sentiment and style transfer. *arXiv preprint arXiv:1804.06437*.
- Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. In *NeurIPS Autodiff Workshop*.
- Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. *SemEval-2014 task 4: Aspect based sentiment analysis*. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland. Association for Computational Linguistics.
- Shrimai Prabhumoye, Yulia Tsvetkov, Ruslan Salakhutdinov, and Alan W Black. 2018. Style transfer through back-translation. *arXiv preprint arXiv:1804.09000*.
- Reid Pryzant, Richard Diehl Martinez, Nathan Dass, Sadao Kurohashi, Dan Jurafsky, and Diyi Yang. 2020. Automatically neutralizing subjective bias in text. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 480–489.
- EFTK SANG. 1999. Representing text chunks. In *Proceedings of EACL'99*, pages 173–179.
- Erik Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the conll-2000 shared task chunking. In *Fourth Conference on Computational Natural Language Learning and the Second Learning Language in Logic Workshop*.
- Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. In *Advances in neural information processing systems*, pages 6830–6841.
- Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2014. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *Workshop at International Conference on Learning Representations*. Iclr.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. Learning to attend via word-aspect associative fusion for aspect-based sentiment analysis. In *Thirty-second AAAI conference on artificial intelligence*.

Youzhi Tian, Zhiting Hu, and Zhou Yu. 2018. Structured content preservation for unsupervised text style transfer. *arXiv preprint arXiv:1810.06526*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Ke Wang, Hang Hua, and Xiaojun Wan. 2019. Controllable unsupervised text attribute transfer via editing entangled latent representation. In *Advances in Neural Information Processing Systems*, pages 11034–11044.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierrette Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface’s transformers: State-of-the-art natural language processing. *ArXiv*, pages arXiv–1910.

Xing Wu, Tao Zhang, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. ”mask and infill”: Applying masked language model to sentiment transfer. *arXiv preprint arXiv:1908.08039*.

Hu Xu, Bing Liu, Lei Shu, and Philip S Yu. 2019. Bert post-training for review reading comprehension and aspect-based sentiment analysis. *arXiv preprint arXiv:1904.02232*.

Jingjing Xu, Xu Sun, Qi Zeng, Xuancheng Ren, Xiaodong Zhang, Houfeng Wang, and Wenjie Li. 2018. Unpaired sentiment-to-sentiment translation: A cycled reinforcement learning approach. *arXiv preprint arXiv:1805.05181*.

Zichao Yang, Zhiting Hu, Chris Dyer, Eric P Xing, and Taylor Berg-Kirkpatrick. 2018. Unsupervised text style transfer using language models as discriminators. In *Advances in Neural Information Processing Systems*, pages 7287–7298.

Zhirui Zhang, Shuo Ren, Shujie Liu, Jianyong Wang, Peng Chen, Mu Li, Ming Zhou, and Enhong Chen. 2018. Style transfer as unsupervised machine translation. *arXiv preprint arXiv:1808.07894*.

# Energy-based Self-attentive Learning of Abstractive Communities for Spoken Language Understanding

Guokan Shang<sup>1,2</sup>, Antoine J.-P. Tixier<sup>1</sup>,  
Michalis Vazirgiannis<sup>1,3</sup>, Jean-Pierre Lorré<sup>2</sup>

<sup>1</sup>École Polytechnique, <sup>2</sup>Linagora, <sup>3</sup>AUEB

## Abstract

Abstractive community detection is an important spoken language understanding task, whose goal is to group utterances in a conversation according to whether they can be jointly summarized by a common abstractive sentence. This paper provides a novel approach to this task. We first introduce a neural contextual utterance encoder featuring three types of self-attention mechanisms. We then train it using the siamese and triplet energy-based meta-architectures. Experiments on the AMI corpus show that our system outperforms multiple energy-based and non-energy based baselines from the state-of-the-art. Code and data are publicly available<sup>1</sup>.

## 1 Introduction

Today, large amounts of digital text are generated by spoken or written conversations, let them be human-human (customer service, multi-party meetings) or human-machine (chatbots, virtual assistants). Such text comes in the form of transcriptions. A transcription is a list of time-ordered text fragments called *utterances*. Abstractive summarization of conversations is an open problem in NLP. Previous work (Mehdad et al., 2013; Oya et al., 2014; Banerjee et al., 2015; Shang et al., 2018) decomposes this task into two subtasks a and b as shown in Fig. 1.

Subtask a, or *Abstractive Community Detection* (ACD), is the focus of this paper. It consists in grouping utterances according to whether they can be jointly summarized by a common abstractive sentence (Murray et al., 2012). Such groups of utterances are called *abstractive communities*. Once they are obtained, an abstractive sentence is generated for each group (subtask b), thus forming the final summary. ACD includes, but is a more

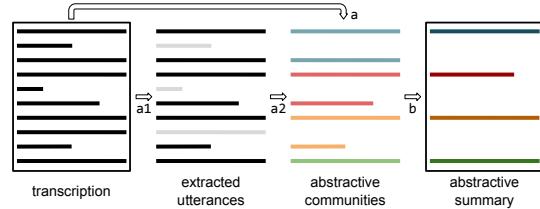


Figure 1: Abstractive summarization of conversations.

general problem than, topic clustering. Indeed, as shown in Fig. 2, communities should capture more complex relationship than simple semantic similarity. Also, two utterances may be part of the same community even if they are not close to each other in the transcription. Finally, a given utterance may belong to more than one community, which results in overlapping groupings (e.g., A and D in Fig. 2), or be a singleton community (B in Fig. 2).

In this paper, we depart from previous work and argue that the ACD subtask should be broken down into two steps, a1 and a2 in Fig. 1. That is, summary-worthy utterances should first be extracted from the transcription (a1), and then, grouped into abstractive communities (a2). This process is more consistent with the way humans treat the summarization task. E.g., during the creation of the AMI corpus (McCowan et al., 2005), annotators were first asked to extract summary-worthy utterances from the transcription, and then to link the selected utterances to the sentences in the abstractive summary (links in Fig. 2), i.e., create communities. Abstractive summaries comprise four sections: ABSTRACT, ACTIONS, PROBLEMS, and DECISIONS. Step a1 plays an important filtering role, since in practice, only a small part of the original utterances are used to construct the abstractive communities (17% on average for AMI). However, this step is closely related to *extractive summarization*, which has been extensively studied in the conversational domain (Murray et al.,

<sup>1</sup>[https://bitbucket.org/guokan\\_shang/abscomm](https://bitbucket.org/guokan_shang/abscomm)

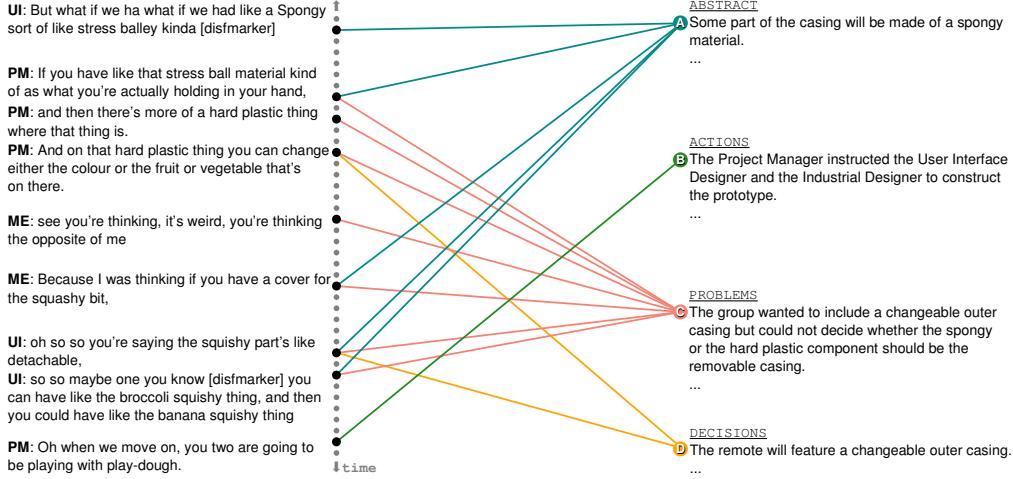


Figure 2: Example of ground truth human annotations from the ES2011c AMI meeting. Successive grey nodes on the left denote utterances in the transcription. Black nodes correspond to the utterances judged important. Sentences (e.g., A, B, C, D) from the abstractive summary are shown on the right. All utterances linked to the same abstractive sentence form one community.

2005; Garg et al., 2009; Tixier et al., 2017).

Rather, we focus in this paper on the rarely explored a2 *utterance clustering* step, which we think is an important spoken language understanding problem, as it plays a crucial role of bridge between two major types of summaries: extractive and abstractive.

## 2 Departure from previous work

Prior work performed ACD either in a supervised (Murray et al., 2012; Mehdad et al., 2013) or unsupervised way (Oya et al., 2014; Banerjee et al., 2015; Singla et al., 2017; Shang et al., 2018).

In the supervised case, Murray et al. (2012) train a logistic regression classifier with handcrafted features to predict extractive-abstractive links, then build an utterance graph whose edges represent the binary predictions of the classifier, and finally apply an overlapping community detection algorithm to the graph. Mehdad et al. (2013) add to the previous approach by building an entailment graph for each community, where edges are entailment relations between utterances, predicted by a SVM classifier trained with handcrafted features on an external dataset. The entailment graph allows less informative utterances to be eliminated from each community.

On the other hand, unsupervised approaches to ACD do not make use of extractive-abstractive links. Oya et al. (2014); Banerjee et al. (2015); Singla et al. (2017) assume that disjoint topic segments (Galley et al., 2003) align with abstractive communities, while Shang et al. (2018) use the classical vector space representation with TF-

IDF weights, and apply  $k$ -means to the LSA-compressed utterance-term matrix.

To sum up, prior ACD methods either train multiple models on different labeled datasets and heavily rely on handcrafted features, or are incapable of capturing the complicated structure of abstractive communities described in the introduction.

Motivated by the recent success of energy-based approaches to similarity learning tasks such as face verification (Schroff et al., 2015) and sentence matching (Mueller and Thyagarajan, 2016), we introduce in this paper a novel utterance encoder, and train it within the siamese (Chopra et al., 2005) and triplet (Hoffer and Ailon, 2015) energy-based meta-architectures. Our final network is able to accurately capture the complexity of abstractive community structure, while at the same time, it is trainable in an end-to-end fashion without the need for human intervention and handcrafted features. Our contributions are threefold: (1) we formalize ACD, a crucial subtask for abstractive summarization of conversations, and publicly release a version of the AMI corpus preprocessed for this subtask, to foster research on this topic, (2) we propose one of the first applications of energy-based learning to spoken language understanding, and (3) we introduce a novel utterance encoder featuring three types of self-attention mechanisms and taking contextual and temporal information into account.

## 3 Energy framework

Energy-Based Modeling (EBM) (LeCun and Huang, 2005; LeCun et al., 2006) is a unified framework that can be applied to many machine learning

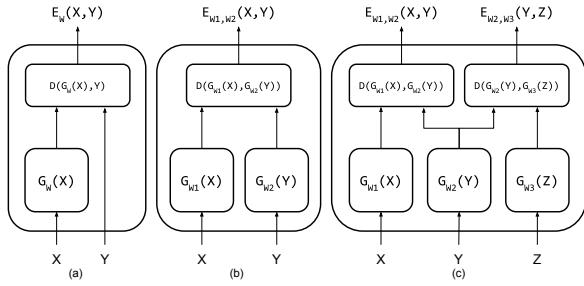


Figure 3: Three EBM architectures. When all  $G$ s and  $W$ s are equal, (b) and (c) correspond to the siamese/triplet cases.

problems. In EBM, an energy function assigns a scalar called *energy* to each pair of random variables  $(X, Y)$ . The energy can be interpreted as the incompatibility between the values of  $X$  and  $Y$ . Training consists in finding the parameters  $W^*$  of the energy function  $E_W$  that, for all  $(X^i, Y^i)$  in the training set  $\mathcal{S}$  of size  $P$ , assign low energy to compatible (correct) combinations and high energy to all other incompatible (incorrect) ones. This is done by minimizing a *loss functional*<sup>2</sup>  $\mathcal{L}$ :

$$W^* = \arg \min_{W \in \mathcal{W}} \mathcal{L}(E_W(X, Y), \mathcal{S}) \quad (1)$$

For a given  $X$ , prediction consists in finding the value of  $Y$  that minimizes the energy.

### 3.1 Single architecture

In the EBM framework, a regression problem can be formulated as shown in Fig. 3a, where the input  $X$  is passed through a regressor model  $G_W$  and the scalar output is compared to the desired output  $Y$  with a dissimilarity measure  $D$  such as the squared error. Here, the energy function is the loss functional to be minimized.

$$\mathcal{L} = \frac{1}{P} \sum_{i=1}^P E_W(X^i, Y^i) = \frac{1}{P} \sum_{i=1}^P \|G_W(X^i) - Y^i\|^2 \quad (2)$$

### 3.2 Siamese architecture

In the regression problem previously described, the dependence between  $X$  and  $Y$  is expressed by a direct mapping  $Y = f(X)$ , and there is a single best  $Y^*$  for every  $X$ . However, when  $X$  and  $Y$  are not in a predictor/predictand relationship but are exchangeable instances of the same family of objects, there is no such mapping. E.g., in paraphrase identification, a sentence may be similar

<sup>2</sup>the loss *functional* is passed the output of the energy function, unlike a loss *function* which is directly fed the output of the model.

to many other ones, or, in language modeling, a given  $n$ -gram may be likely to be followed by many different words.

Thereby, Lecun et al. (2006) introduced EBM for *implicit regression* or *constraint satisfaction* (see Fig. 3b), in which a constraint that  $X$  and  $Y$  must satisfy is defined, and the energy function measures the extent to which that constraint is violated:

$$E_{W_1, W_2}(X, Y) = D(G_{W_1}(X), G_{W_2}(Y)) \quad (3)$$

where  $G_{W_2}$  and  $G_{W_1}$  are two functions parameterized by  $W_1$  and  $W_2$ . When  $G_{W_1} = G_{W_2}$  and  $W_1 = W_2$ , we obtain the well-known siamese architecture (Bromley et al., 1994; Chopra et al., 2005), which has been applied with success to many tasks, including sentence similarity (Mueller and Thyagarajan, 2016).

Here, the constraint is determined by a collection-level set of binary labels  $\{C^i\}_{i=1}^P$ .  $C^i = 0$  indicates that  $(X^i, Y^i)$  is a *genuine* pair (e.g., two paraphrases), while  $C^i = 1$  indicates that  $(X^i, Y^i)$  is an *impostor* pair (e.g., two sentences with different meanings).

The function  $G_W$  projects objects into an embedding space such that the defined dissimilarity measure  $D$  (e.g., Euclidean distance) in that space reflects the notion of dissimilarity in the input space. Thus, the energy function can be seen as a metric to be learned.

We experiment with deep neural network encoders as  $G_W$ , and, following (Mueller and Thyagarajan, 2016), we adopt the exponential negative Manhattan distance as dissimilarity measure and the mean squared error as loss functional:

$$E_W(X, Y) = 1 - \exp(-\|G_W(X) - G_W(Y)\|_1) \quad (4)$$

$$\mathcal{L} = \frac{1}{P} \sum_{i=1}^P \|E_W(X^i, Y^i) - C^i\|^2 \quad (5)$$

### 3.3 Triplet architecture

The triplet architecture (Schroff et al., 2015; Hoffer and Ailon, 2015; Wang et al., 2014), as can be seen in Fig. 3c, is a direct extension of the siamese architecture that takes as input a triplet  $(X, Y, Z)$  in lieu of a pair  $(X, Y)$ .  $X$ ,  $Y$ , and  $Z$  are referred to as the *positive*, *anchor*, and *negative* objects, respectively.  $X$  and  $Y$  are similar, while both being dissimilar to  $Z$ . Learning consists in jointly minimizing the positive-anchor energy  $E_W(X^i, Y^i)$  while maximizing the anchor-negative energy  $E_W(Y^i, Z^i)$ .

Here, we use the *softmax triplet loss* (Hoffer and Ailon, 2015) as our loss functional:

$$\mathcal{L} = \frac{1}{2P} \sum_{i=1}^P (\|ne^+ - 0\|^2 + \|ne^- - 1\|^2) \quad (6)$$

$$ne^+ = \frac{e^{E_W(X^i, Y^i)}}{e^{E_W(X^i, Y^i)} + e^{E_W(Y^i, Z^i)}} \quad (7)$$

$$ne^- = \frac{e^{E_W(Y^i, Z^i)}}{e^{E_W(X^i, Y^i)} + e^{E_W(Y^i, Z^i)}} \quad (8)$$

where  $ne$  stands for normalized energy, and the dissimilarity measure is the Euclidean distance, i.e.,  $E_W(X^i, Y^i) = \|G_W(X^i) - G_W(Y^i)\|_2$ . Essentially, the softmax triplet loss is the mean squared error between the normalized energy vector  $[ne^+, ne^-]$  and  $[0, 1]$ . We justify our choice of loss functionals in App. F.

### 3.4 Sampling procedures

We sample tuples from the ground truth abstractive communities to train our utterance encoder  $G_W$  under the siamese and triplet meta-architectures as follows.

**Pair sampling.** All utterances belonging to the same community are paired as genuine pairs, while impostor pairs are any two utterances coming from different communities.

**Triplet sampling.** Utterances from the same community provide positive and anchor items, while the negative item is taken from any other community.

## 4 Proposed utterance encoder

**Notation.** The *time*  $t$  (as superscript) denotes the position of a given utterance in the conversation of length  $T$ , and the *position*  $i$  (as subscript) denotes the position of a token within a given utterance of length  $N$ . E.g.,  $\mathbf{u}_1^t$  is the representation of the first token of  $\mathbf{U}^t$ , the  $t^{th}$  utterance in the transcription.

**Word encoder.** As shown in the upper right corner of Fig. 4, we obtain  $\mathbf{u}_i^t$  by concatenating the pre-trained vector of the corresponding token with the discourse features of  $\mathbf{U}^t$  (role, position and dialogue act), and passing the resulting vector to a dense layer.

**Utterance encoder.** As shown in the center of Fig. 4, we represent  $\mathbf{U}^t$  as a sequence of  $N$   $d$ -dimensional token representations  $\{\mathbf{u}_1^t, \dots, \mathbf{u}_N^t\}$ . In addition, because there is a strong time dependence between utterances (see Fig. 2), we inform the model about the preceding and following utterances when encoding  $\mathbf{U}^t$ . To accomplish this, we prepend (resp. append) to  $\mathbf{U}^t$  a context vector containing information about the

previous (resp. next) utterances, finally obtaining  $\mathbf{U}^t = \{\mathbf{u}_{\text{pre}}^t, \mathbf{u}_1^t, \dots, \mathbf{u}_N^t, \mathbf{u}_{\text{post}}^t\} \in \mathbb{R}^{(N+2) \times d}$ . We then use a non-stacked bidirectional Recurrent Neural Network (RNN) with Gated Recurrent Units (GRU) (Cho et al., 2014) to transform  $\mathbf{U}^t$  into a sequence of annotations  $\mathbf{H}^t \in \mathbb{R}^{(N+2) \times 2d}$ .

In practice, the pre and post-context vectors indirectly initialize the left-to-right and right-to-left RNNs with information about the utterances preceding and following  $\mathbf{U}^t$ . This is similar in spirit to the warm-start method of Wang et al. (2017), that directly initializes the hidden states of the RNNs with the context vectors.

**Self-attention.** We then pass the annotations  $\mathbf{H}^t$  to a self-attention mechanism (Vaswani et al., 2017; Lin et al., 2017). More precisely,  $\mathbf{H}^t$  go through a dense layer and the output is compared via dot product with a trainable vector  $\mathbf{u}_\gamma$ , initialized randomly. Then, a probability distribution over the  $N + 2$  tokens in  $\mathbf{U}^t$  is obtained via a softmax:

$$\boldsymbol{\gamma}^t = \text{softmax}(\mathbf{u}_\gamma \cdot \tanh(\mathbf{W}_\gamma \mathbf{H}^t)) \quad (9)$$

The attentional vector for  $\mathbf{U}^t$  is finally computed as a weighted sum of its annotations, and, as shown in Fig. 4, is finally passed to a dense layer to obtain the utterance embedding  $\mathbf{u}^t \in \mathbb{R}^{df}$ :

$$\mathbf{u}^t = \text{dense}\left(\sum_{i=1}^{N+2} \gamma_i^t \mathbf{h}_i^t\right) \quad (10)$$

$\mathbf{u}_\gamma$  can be interpreted as a learned representation of the “ideal word”, on average. The more similar a token vector is to this representation, the more attention the model pays to the token.

**Context encoder: level 1.** The pre and post-context vectors that we prepend and append to  $\mathbf{U}^t$  are obtained by aggregating information from the  $C$  utterances preceding and following  $\mathbf{U}^t$ :

$$\mathbf{u}_{\text{pre}}^t \leftarrow \text{aggregate}_{\text{pre}}(\{\mathbf{U}^{t-C}, \dots, \mathbf{U}^{t-1}\}) \quad (11)$$

$$\mathbf{u}_{\text{post}}^t \leftarrow \text{aggregate}_{\text{post}}(\{\mathbf{U}^{t+1}, \dots, \mathbf{U}^{t+C}\}) \quad (12)$$

where  $C$ , the context size, is a hyperparameter. Since  $\mathbf{u}_{\text{pre}}^t$  and  $\mathbf{u}_{\text{post}}^t$  will become part of utterance  $\mathbf{U}^t$  which is a sequence of token vectors, and fed to the RNN, we need them to live in the same space as any other token vector. This forbids the use of any nonlinear or dimension-changing transformation in aggregate, such as convolutional or recurrent operations. Therefore, we use self-attention only. More precisely, we propose a two-level hierarchical architecture that makes use of a different type of self-attention at each level (see left part of Fig. 4). The pre and post-context encoders share the exact

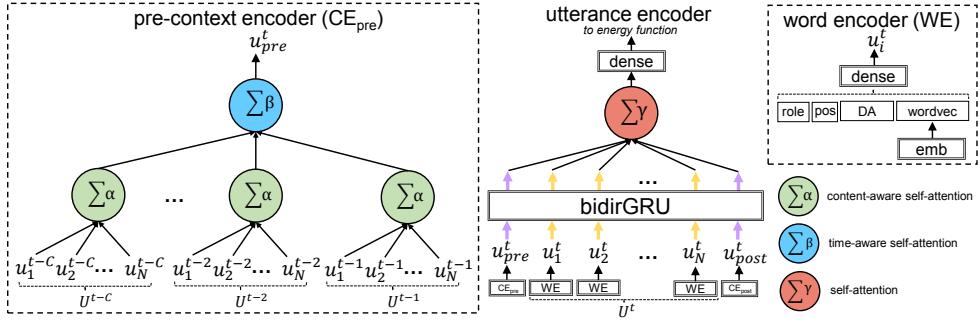


Figure 4: Our proposed utterance encoder  $G_W$ . Only the pre-context encoder is shown.  $C$  is the context size.

same architecture, so we only describe the pre-context encoder in what follows.

*Content-aware self-attention.* At level 1, we apply the same attention mechanism to each utterance in  $\{\mathbf{U}^{t-C}, \dots, \mathbf{U}^{t-1}\}$ . E.g., for  $\mathbf{U}^{t-1}$ :

$$\boldsymbol{\alpha}^{t-1} = \text{softmax} \left( \mathbf{u}_\alpha \cdot \tanh \left( \mathbf{W}_\alpha \mathbf{U}^{t-1} + \mathbf{W}' \sum_{i=1}^N \mathbf{u}_i^t \right) \right) \quad (13)$$

This mechanism is the same as in Eq. 9, except for two differences. First, we operate directly on the matrix of token vectors of the previous utterance  $\mathbf{U}^{t-1}$  rather than on RNN annotations. Second, there is an extra input that consists of the element-wise sum of the token vectors of the current utterance  $\mathbf{U}^t$ . The latter modification is inspired by the coverage vectors used in translation and summarization to address under(over)-translation and repetition, e.g., (Tu et al., 2016; See et al., 2017). In our case, we hope that by letting the model know about the tokens in the current utterance  $\mathbf{U}^t$ , it will be able to extract complementary -rather than redundant- information from its context, and thus produce a richer embedding.

To recapitulate, the content-aware attention mechanism transforms the sequence of utterance matrices  $\{\mathbf{U}^{t-C}, \dots, \mathbf{U}^{t-1}\} \in \mathbb{R}^{C \times N \times d}$  into a sequence of vectors  $\{\mathbf{u}^{t-C}, \dots, \mathbf{u}^{t-1}\} \in \mathbb{R}^{C \times d}$ . These vectors are then aggregated into a single pre-context vector  $\mathbf{u}_{pre}^t \in \mathbb{R}^d$  as described next.

Note that since there is no inherent difference between preceding and following utterances<sup>3</sup>, we use the same content-aware self-attention mechanism for the pre and post contexts. This also gives us a more parsimonious and faster model. One should note, however, that the pre and post-context encoders still differ in terms of their time-aware attention mechanisms at level 2, described next.

<sup>3</sup>indeed, the latter become the former as we slide the window over the transcription

**Context encoder: level 2.** As can be seen in Fig. 2, two utterances close to each other in time are much more likely to be related (e.g., adjacency pair, elaboration...) than any two randomly selected utterances. To enable our model to capture such time dependence, we use the trainable universal time-decay attention mechanism of Su et al. (2018).

*Time-aware self-attention.* The mechanism combines three types of time-decay functions via weights  $w_i$ . The attentional coefficient for  $\mathbf{u}^{t-1}$  is:

$$\begin{aligned} \beta^{t-1} &= w_1 \beta^{\text{conv}^{t-1}} + w_2 \beta^{\text{lin}^{t-1}} + w_3 \beta^{\text{conc}^{t-1}} \\ &= \frac{w_1}{a(d^{t-1})^b} + w_2 [ed^{t-1} + k]^+ + \frac{w_3}{1 + (\frac{d^{t-1}}{D_0})^l} \end{aligned} \quad (14)$$

where  $[*]^+ = \max(*, 0)$  (ReLU),  $d^{t-1}$  is the offset between the positions of  $\mathbf{U}^{t-1}$  and  $\mathbf{U}^t$ , i.e.,  $d^{t-1} = |t - (t-1)| = 1$ , and the  $w_i$ 's,  $a$ ,  $b$ ,  $e$ ,  $k$ ,  $D_0$ , and  $l$  are scalar parameters learned during training.

The convex (conv), linear (lin), and concave (conc) terms respectively assume the strength of dependence to weaken rapidly, linearly, and slowly, as the distance in time increases. The post-context mechanism can be obtained by symmetry. It has different parameters.

## 5 Community detection

Once the utterance encoder  $G_W$  presented in Section 4 has been trained within the siamese or triplet meta-architecture presented in Section 3, it is used to project the summary-worthy utterances from a given test transcription to a compact embedding space. We assume that if training was successful, the distance in that space encodes community structure, so that a basic clustering algorithm such as  $k$ -means (MacQueen, 1967) is enough to capture it. However, since we need to detect overlapping communities, we use a probabilistic version of  $k$ -means, the Fuzzy c-Means (FCM) algorithm (Bezdek et al.,

1984). FCM returns a probability distribution over all communities for each utterance. More details are provided in App. E.

## 6 Experimental setup

### 6.1 Dataset

We experiment on the AMI corpus (McCowan et al., 2005), with the manual annotations v1.6.2. The corpus contains data for more than 100 meetings, in which participants play 4 roles within a design team whose task is to develop a prototype of TV remote control. Each meeting is associated with the annotations described in the introduction and shown in Fig. 2. There are 2368 unique abstractive communities in total, whose statistics are shown in Table 1. We adopt the officially suggested *scenario-only partition*<sup>4</sup>, which provides 97, 20, and 20 meetings respectively for training, validation and testing. We use manual transcriptions, and do not apply any particular preprocessing except filtering out specific ASR tags, such as `vocalsound`.

type	abstract	action	problem	decision	total
unique	1147	247	380	594	2368
disjoint	528	124	69	45	766
overlapping	349	17	163	149	678
singletone	49	162	38	244	493

Table 1: Statistics of abstractive communities.

### 6.2 Baselines

Full baseline details are provided in App. B.

- **Encoders.** First, we evaluate our utterance encoder against two encoders that are trained within the energy framework: (1) **LD** (Lee and Dernoncourt, 2016), a sequential sentence encoder developed for dialogue act classification; and (2) **HAN** (Yang et al., 2016), a hierarchical self-attentive network for document embedding. Note that to be fair, we ensure that both LD and HAN have access to context (see details in App. B).

We also compare our full pipeline against unsupervised and supervised systems.

- **Unsupervised systems.** In (1) **tf-idf**, we combine the TF-IDF vectors of the current utterance and the context utterances, each concatenated with their discourse features, and apply FCM. In (2) **w2v**, we repeat the same approach with the word2vec centroids of the words in each utterance. We also compare our full pipeline against **LCseg** (Galley et al., 2003), a lexical-cohesion based topic

segmenter that directly clusters utterances without computing embeddings.

- **Supervised systems.** Finally, here, we use an approach similar to that of Murray et al. (2012). More precisely, we train a MLP to learn abstractive links between utterances, and then apply the CONGA community detection algorithm to the utterance graph.

We also considered 4 variants of our model: (1) **CA-S**: we replace the time-aware self-attention mechanism of the context encoder with basic self-attention. (2) **S-S**: we replace both the content-aware and the time-aware self-attention mechanisms of the context encoder with basic self-attention. (3) **(0,0)**: our model, without using the contextual encoder. (4) **(3,0)**: our model, using only pre-context, with a small window of 3, to enable fair comparison with the LD baseline.

### 6.3 Training details

**Word encoder.** Discourse features consist of two one-hot vectors of dimensions 4 and 16, respectively for speaker role and dialogue act. The positional feature is a scalar in [0, 1], indicating the normalized position of the utterance in the transcription. We used the pre-trained vectors learned on the Google News corpus with word2vec by (Mikolov et al., 2013), and randomly initialized out-of-vocabulary words (1645 out of 12412). As a preprocessing step, we reduced the dimensionality of the pre-trained word vectors from 300 to 21 with PCA, in order to give equal importance to discourse and textual features. In the end, tokens are thus represented by a  $d = 42$ -dimensional vector.

**Layer sizes.** For our model, and the LD and HAN baselines, we set  $d_f = 32$  (output dimension of the final dense layer).

**LD.** We set  $d_1=3$  and  $d_2=0$ , which is very close to (2,0), the best configuration reported in the original paper.

**HAN.** Again, for the sake of fairness, we give the HAN baseline access to contextual information, by feeding it the current utterance surrounded by the  $C_b$  preceding and  $C_b$  following utterances in the transcription, where  $C_b$  denotes the best context size reported in Section 7.

**Training details.** The exact same token representations and settings were used for our model, its variants, and the baselines. Models were trained on the training set for 30 epochs with the Adam (Kingma and Ba, 2015) optimizer. The best epoch

<sup>4</sup><http://groups.inf.ed.ac.uk/ami/corpus/datasets.shtml>

was selected as the one associated with the lowest validation loss. Batch size and dropout (Srivastava et al., 2014) were set to 16 and 0.5. Dropout was applied to the word embedding layer only. To account for randomness, we average results over 10 runs. Also, following (Hoffer and Ailon, 2015; Liu et al., 2019), we use a different, small subset of all possible triplets for training at each epoch (more precisely, 15594 triplets). This intelligently maximizes data usage while preventing overfitting. To enable fair comparison with the siamese approach, 15594 genuine and 15594 impostor pairs were sampled at the beginning of each epoch, since we consider that one triplet essentially equates one genuine pair and one impostor pair.

**Performance evaluation.** We evaluate performance at the distance and the clustering level, using respectively precision, recall, and F1 score at  $k$ , and the Omega Index (Collins and Dent, 1988) following the previous ACD work of Murray et al. (2012). For P, R, and F1, we evaluate the quality of the ranking of the closest utterances to a given query utterance. We use a fixed  $k=10$  and also a variable  $k$  (denoted as  $k=v$ ), where  $k$  is equal to the size of the community of the query utterance minus one. In that case,  $P=R=F1$ . More details and examples are given in Appendices C and D.

For the Omega Index, we report results with a fixed number of communities  $|Q|=11$ , and also a variable  $|Q|$  ( $|Q|=v$ ), where  $|Q|$  is equal to the number of ground truth communities. More details and examples are given in App. C.

Due to the stochastic nature of the FCM algorithm, we select the run yielding the smallest objective function value over 20 runs.

## 7 Results

**Context sizes.** Larger contexts bring richer information, but increase the risk of considering unrelated utterances. Using our proposed encoder within the triplet meta-architecture, we tried different values of  $C$  on the validation set, under two settings:  $(\text{pre}, \text{post}) = (C, 0)$ , and  $(\text{pre}, \text{post}) = (C, C)$ . Results are shown in Fig. 5. We can observe that increasing  $C$  always brings improvement, with diminishing returns. Results also clearly show that considering the following utterances in addition to the preceding ones is useful. Note that the curves look similar for  $F1@k = 10$ . In the end, we selected (11,11) as our best context sizes.

**Quantitative results.** Final test set results are

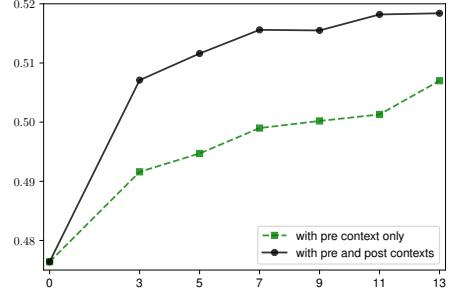


Figure 5: Impact of context size on the validation  $P@k = v$ , for our model trained within the triplet meta-architecture.

shown in Table 2. All variants of our model significantly outperform LD. While HAN is much stronger than LD, our model and its variants using best context sizes manage to outperform it everywhere, except in the siamese/P@k=v case (row j). One of the reasons for the superiority of our utterance encoder is probably that it considers contextual information *while encoding* the current utterance, while HAN and LD take as input the context utterances together with the current utterance, without distinguishing between them. Moreover, we use an attention mechanism dedicated to temporality, whereas HAN is only able to capture an implicit notion of time through the use of recurrence (RNN), and LD, with its dense layers, completely ignores it. Also, all variants of our model using best context sizes (11,11) outperform the ones using reduced (3,0) or no (0,0) context, regardless of the meta-architecture. This confirms the value added by our context encoder.

For siamese, our model outperforms its two variants (CA-S and S-S) for all metrics, indicating that both the content-aware and the time-aware self-attention mechanisms are useful. However, it is interesting to note that when training under the triplet configuration, the CA-S variant of our model is better, suggesting that in that case, the content-aware mechanism is beneficial, but the time-aware one is not.

LCseg (row m) and tf-idf (11,11) (row n3) are the best of all (un)supervised baseline systems, but both perform significantly worse than all energy-based approaches, highlighting that training with the energy framework is beneficial. In terms of Omega Index, supervised baseline systems are logically better than unsupervised ones.

w2v generally outperforms tf-idf when there is no context (rows k1,l1,n1,o1) or short context (k2,l2,n2,o2), but not with large contexts (k3,l3,n3,o3). Results also show that overall, using

			(pre, post)	P	P	R	F1	Omega index $\times 100$	
			@ $k = v$	@ $k = 10$				$Q  = v$	$Q  = 11$
Triplet	a1)	our model	(0, 0)	54.59	46.05	62.45	43.18	49.09	48.81
	a2)	our model	(3, 0)	55.17	46.17	62.80	43.25	49.78	49.70
	a3)	our model	(11, 11)	58.58	46.73	63.82	43.83	49.90	49.28
	b)	our model (CA-S)	(11, 11)	<b>59.52*</b>	<b>46.98*</b>	<b>64.01*</b>	<b>44.06*</b>	<b>50.11</b>	49.73
	c)	our model (S-S)	(11, 11)	58.96	46.81	63.65	43.87	49.59	<b>49.88</b>
	d)	LD	(3, 0)	52.04	44.82	60.41	41.82	48.70	48.14
Siamese	e)	HAN	(11, 11)	58.72	45.76	62.60	42.89	49.32	48.88
	f1)	our model	(0, 0)	53.01	45.10	60.97	42.12	50.56	49.65
	f2)	our model	(3, 0)	53.78	45.54	61.33	42.48	51.01	50.00
	f3)	our model	(11, 11)	56.64	<b>46.47</b>	<b>62.54</b>	<b>43.40</b>	<b>52.44*</b>	<b>51.88*</b>
	g)	our model (CA-S)	(11, 11)	56.46	46.08	61.92	43.02	51.60	50.98
	h)	our model (S-S)	(11, 11)	55.68	45.64	61.17	42.53	52.26	51.11
	i)	LD	(3, 0)	52.13	44.83	60.85	41.86	51.18	50.70
Unsupervised	j)	HAN	(11, 11)	<b>58.54</b>	45.72	61.55	42.74	50.51	49.82
	k1)	tf-idf	(0, 0)	29.28	26.67	34.69	24.19	13.12	13.66
	k2)	tf-idf	(3, 0)	34.77	30.27	40.83	27.79	10.22	10.17
	k3)	tf-idf	(11, 11)	<b>58.94</b>	43.94	61.36	41.45	38.09	39.47
	l1)	w2v	(0, 0)	29.02	27.46	37.39	25.11	13.89	13.50
	l2)	w2v	(3, 0)	34.11	29.92	39.55	27.32	10.61	10.77
	l3)	w2v	(11, 11)	58.30	<b>44.08</b>	<b>61.59</b>	<b>41.59</b>	37.75	38.28
Supervised	m)	LCSeg	-	-	-	-	-	<b>38.98</b>	<b>41.57</b>
	n1)	tf-idf	(0, 0)	-	-	-	-	25.04	25.14
	n2)	tf-idf	(3, 0)	-	-	-	-	27.33	26.95
	n3)	tf-idf	(11, 11)	-	-	-	-	<b>45.26</b>	<b>44.91</b>
	o1)	w2v	(0, 0)	-	-	-	-	25.32	25.25
	o2)	w2v	(3, 0)	-	-	-	-	29.14	29.02
	o3)	w2v	(11, 11)	-	-	-	-	43.31	43.08

Table 2: Results (averaged over 10 runs). \*: best score per column. **Bold**: best score per section. -: does not apply as the method does not produce utterance embeddings.

larger contexts always brings improvement.

**Qualitative results.** We visualize in App. A that the three self-attention mechanisms behave in a cooperative manner to produce a meaningful utterance representation. We also inspect the closest utterances to a given query utterance in App. D. We also visualize the time-aware self-attention coefficients in Fig. 6, and find that interestingly, the distributions over the pre and post-context are not symmetric. Indeed, only the utterances immediately following  $\mathbf{U}^t$  ( $t+1 \rightarrow t+5$ ) seem to matter, while the attention weights are much more uniform across the utterances preceding  $\mathbf{U}^t$ . This suggests that in dialogues, considering a long history of preceding utterances helps understanding the current one. Overall, the three terms (see Eq. 14) altogether do produce a universal function that decreases as time distance increases, which is in accordance with intuition.

**Simplified task.** Finally, we also experimented on a much simpler task, where only the communities of type ABSTRACT were considered. This makes ACD much simpler, because most of the overlapping communities are of the other types (see Table 1). For this simplified task, we have 1147 unique

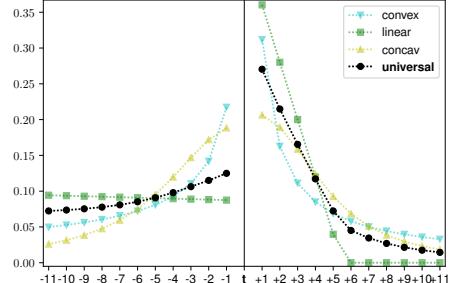


Figure 6: Normalized time-aware self-attention weights for pre and post-contexts, averaged over 10 runs.

communities, of which 78.99% are disjoint. our model achieves 72.09 in terms of  $P@k = v$  and 55.67 in terms of Omega Index when  $|Q| = v$ .  $P, R, F1@k = 15$  are respectively equal to 55.07, 74.37, and 54.00, and the Omega Index is 54.30 when  $|Q| = 8$ .

## 8 Conclusion

This paper proposes one of the first applications of energy-based learning to ACD. Using the siamese and triplet meta-architectures, we showed that our novel contextual utterance encoder learns better distance and communities than state-of-the-art competitors.

## Acknowledgments

This research was supported in part by the OpenPaaS::NG and LinTo (Lorré et al., 2019) projects.

## References

- Siddhartha Banerjee, Prasenjit Mitra, and Kazunari Sugiyama. 2015. [Generating abstractive summaries from meeting transcripts](#). In *Proceedings of the 2015 ACM Symposium on Document Engineering, DocEng '15*, pages 51–60, New York, NY, USA.
- James C. Bezdek, Robert Ehrlich, and William Full. 1984. [Fcm: The fuzzy c-means clustering algorithm](#). *Computers & Geosciences*, 10(2):191 – 203.
- Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. 1994. [Signature verification using a "siamese" time delay neural network](#). In J. D. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems 6*, pages 737–744. Morgan-Kaufmann.
- Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. [Learning phrase representations using RNN encoder–decoder for statistical machine translation](#). In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. ACL.
- S. Chopra, R. Hadsell, and Y. LeCun. 2005. [Learning a similarity metric discriminatively, with application to face verification](#). In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546 vol. 1.
- Linda M. Collins and Clyde W. Dent. 1988. [Omega: A general formulation of the rand index of cluster recovery suitable for non-disjoint solutions](#). *Multivariate Behavioral Research*, 23(2):231–242. PMID: 26764947.
- Michel Galley, Kathleen R. McKeown, Eric Fosler-Lussier, and Hongyan Jing. 2003. [Discourse segmentation of multi-party conversation](#). In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 562–569, Sapporo, Japan. ACL.
- Nikhil Garg, Benoît Favre, Korbinian Riedhammer, and Dilek Hakkani-Tür. 2009. [Clusterrank: a graph based method for meeting summarization](#). In *INTERSPEECH 2009, 10th Annual Conference of the International Speech Communication Association, Brighton, United Kingdom, September 6-10, 2009*, pages 1499–1502. ISCA.
- Steve Gregory. 2007. An algorithm to find overlapping community structure in networks. In *Knowledge Discovery in Databases: PKDD 2007*, pages 91–102, Berlin, Heidelberg. Springer Berlin Heidelberg.
- Elad Hoffer and Nir Ailon. 2015. [Deep metric learning using triplet network](#). In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings*.
- Diederik P. Kingma and Jimmy Ba. 2015. [Adam: A method for stochastic optimization](#). In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.
- Yann Lecun, Sumit Chopra, Raia Hadsell, Marc'Aurelio Ranzato, and Fu Jie Huang. 2006. [A tutorial on energy-based learning](#). In *Predicting structured data*. MIT Press.
- Yann LeCun and Fu Jie Huang. 2005. [Loss functions for discriminative training of energy-based models](#). In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics, AISTATS 2005, Bridgetown, Barbados, January 6-8, 2005*. Society for Artificial Intelligence and Statistics.
- Ji Young Lee and Franck Dernoncourt. 2016. [Sequential short-text classification with recurrent and convolutional neural networks](#). In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 515–520. ACL.
- Zhouhan Lin, Minwei Feng, Cícero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. [A structured self-attentive sentence embedding](#). In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.
- Jinchao Liu, Stuart J. Gibson, James Mills, and Margarita Osadchy. 2019. [Dynamic spectrum matching with one-shot learning](#). *Chemometrics and Intelligent Laboratory Systems*, 184:175 – 181.
- Jean-Pierre Lorré, Isabelle Ferrané, Francisco Madrigal, Michalis Vazirgiannis, and Christophe Bourguignat. 2019. [Linto: Assistant vocal open-source respectueux des données personnelles pour les réunions d’entreprise](#). *APIA*, page 63.
- J. MacQueen. 1967. [Some methods for classification and analysis of multivariate observations](#). In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, pages 281–297, Berkeley, Calif. University of California Press.
- Iain McCowan, Jean Carletta, W Kraaij, S Ashby, S Bourban, M Flynn, M Guillemot, T Hain, J Kadlec, V Karaikos, et al. 2005. [The ami meeting corpus](#). In *Proceedings of the 5th International Conference on Methods and Techniques in Behavioral Research*, volume 88.

- Yashar Mehdad, Giuseppe Carenini, Frank Tompa, and Raymond T. NG. 2013. *Abstractive meeting summarization with entailment and fusion*. In *Proceedings of the 14th European Workshop on Natural Language Generation*, pages 136–146. ACL.
- Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013. *Exploiting similarities among languages for machine translation*. *CoRR*, abs/1309.4168.
- Jonas Mueller and Aditya Thyagarajan. 2016. *Siamese recurrent architectures for learning sentence similarity*. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA.*, pages 2786–2792. AAAI Press.
- Gabriel Murray, Giuseppe Carenini, and Raymond Ng. 2012. *Using the omega index for evaluating abstractive community detection*. In *Proceedings of Workshop on Evaluation Metrics and System Comparison for Automatic Summarization*, pages 10–18, Montréal, Canada. ACL.
- Gabriel Murray, Steve Renals, and Jean Carletta. 2005. *Extractive summarization of meeting recordings*. In *INTERSPEECH 2005 - Eurospeech, 9th European Conference on Speech Communication and Technology, Lisbon, Portugal, September 4-8, 2005*, pages 593–596. ISCA.
- Paul Neculoiu, Maarten Versteegh, and Mihai Rotaru. 2016. *Learning text similarity with siamese recurrent networks*. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 148–157. ACL.
- Tatsuro Oya, Yashar Mehdad, Giuseppe Carenini, and Raymond Ng. 2014. *A template-based abstractive meeting summarization: Leveraging summary and source text relationships*. In *Proceedings of the 8th International Natural Language Generation Conference (INLG)*, pages 45–53. ACL.
- F. Schroff, D. Kalenichenko, and J. Philbin. 2015. *Facenet: A unified embedding for face recognition and clustering*. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 00, pages 815–823.
- Veit Schwämmele and Ole Nørregaard Jensen. 2010. *A simple and fast method to determine the parameters for fuzzy c-means cluster analysis*. *Bioinformatics*, 26(22):2841–2848.
- Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. *Get to the point: Summarization with pointer-generator networks*. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083. ACL.
- Guokan Shang, Wensi Ding, Zekun Zhang, Antoine Tixier, Polykarpos Meladianos, Michalis Vazirgiannis, and Jean-Pierre Lorré. 2018. *Unsupervised abstractive meeting summarization with multi-sentence compression and budgeted submodular maximization*. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 664–674. ACL.
- Karan Singla, Evgeny Stepanov, Ali Orkan Bayer, Giuseppe Carenini, and Giuseppe Riccardi. 2017. *Automatic community creation for abstractive spoken conversations summarization*. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 43–47. ACL.
- Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. *Dropout: A simple way to prevent neural networks from overfitting*. *Journal of Machine Learning Research*, 15:1929–1958.
- Shang-Yu Su, Pei-Chieh Yuan, and Yun-Nung Chen. 2018. *How time matters: Learning time-decay attention for contextual spoken language understanding in dialogues*. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2133–2142, New Orleans, Louisiana. ACL.
- Antoine Tixier, Polykarpos Meladianos, and Michalis Vazirgiannis. 2017. *Combining graph degeneracy and submodularity for unsupervised extractive summarization*. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 48–58. ACL.
- Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. *Modeling coverage for neural machine translation*. *arXiv preprint arXiv:1601.04811*.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. *Attention is all you need*. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 6000–6010.
- Jiang Wang, Yang Song, Thomas Leung, Chuck Rosenberg, Jingbin Wang, James Philbin, Bo Chen, and Ying Wu. 2014. *Learning fine-grained image similarity with deep ranking*. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR ’14*, pages 1386–1393, Washington, DC, USA. IEEE Computer Society.
- Longyue Wang, Zhaopeng Tu, Andy Way, and Qun Liu. 2017. *Exploiting cross-sentence context for neural machine translation*. *arXiv preprint arXiv:1704.04347*.
- Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. *Hierarchical attention networks for document classification*. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489. ACL.

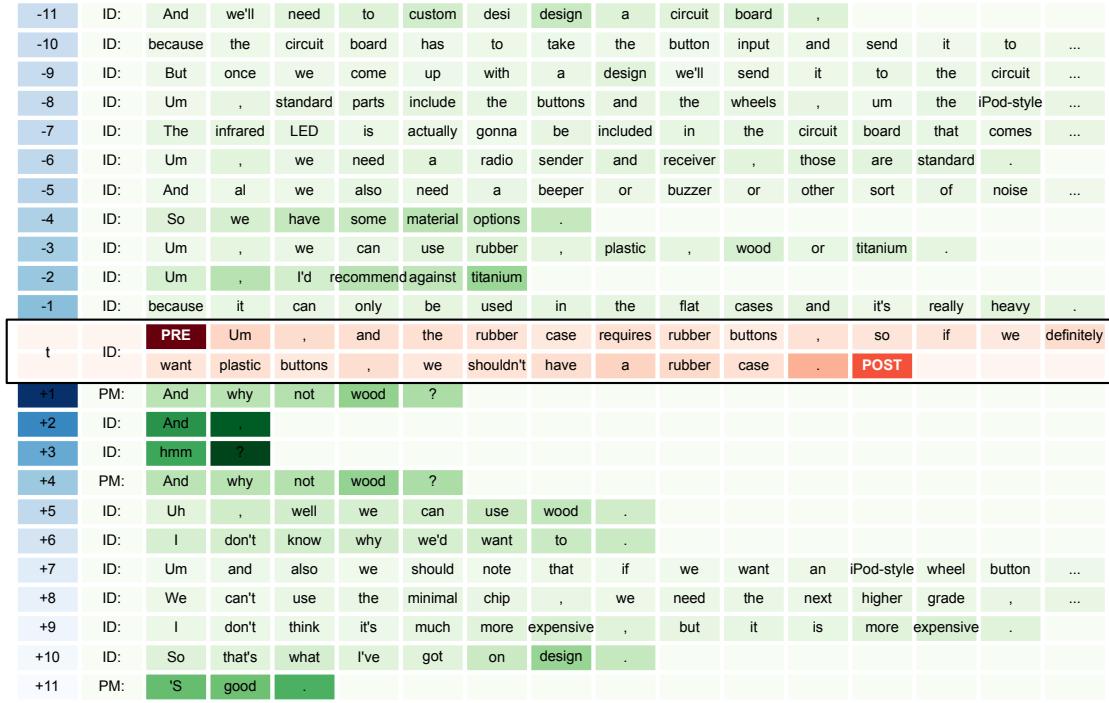


Figure 7: Visualization of attention distributions around an utterance from the ES2011c meeting. Some utterances are truncated for readability.

## Appendices

### A Attention visualization

The aim of this section is to show, with an example, what the three self-attention mechanisms pay attention to while encoding the current utterance  $\mathbf{U}^t$  (here, an utterance from the ES2011c *validation* meeting). Fig. 7 shows the attention distributions over  $\mathbf{U}^t$  (highlighted by the black frame), and over its pre-context  $\{\mathbf{U}^{t-1}, \dots, \mathbf{U}^{t-11}\}$  and post-context  $\{\mathbf{U}^{t+1}, \dots, \mathbf{U}^{t+11}\}$  utterances. We use three colors that are consistent with the ones used in Fig. 4 to denote the three different attention mechanisms: green for content-aware ( $\alpha$ ), blue for time-aware ( $\beta$ ), and red for basic self-attention ( $\gamma$ ). Remember that  $\alpha$  and  $\beta$  are both in the context encoder, while  $\gamma$  is in the utterance encoder. Color shades indicate attention intensity (the darker, the stronger).

We can observe in Fig. 7 that:

- The content-aware self-attention mechanism  $\alpha$  (green) focuses on the informative and complementary words in the contexts that are central to understanding the utterance at time  $t$ , such as: “custom”, “design” from  $\mathbf{U}^{t-11}$ , “material” from  $\mathbf{U}^{t-4}$ , “recommend”, “titanium” from  $\mathbf{U}^{t-2}$ , “wood” from  $\mathbf{U}^{t+1}$ , etc.

- The time-aware self-attention mechanism  $\beta$  (blue) places more importance over the context utterances that are close to  $\mathbf{U}^t$ , i.e., the importance decreases when the time distance increases. However, the patterns are different for the pre and post-contexts (see Fig. 6).

- The self-attention mechanism  $\gamma$  (red) focuses mainly on the special pre-context token PRE, meaning that the pre-context is more important than the post-context in the example considered. Generally speaking, the pre and post-context tokens contain richer information than any token from the current utterance, as the context tokens originate from the fusion of  $\{\mathbf{U}^{t-11}, \dots, \mathbf{U}^t, \dots, \mathbf{U}^{t+11}\}$ . It is thus possible that the utterance encoder has learned to always pay more attention to these information-rich tokens than to any regular token.

- It is also interesting to note that considerable attention is being paid to punctuation marks. This makes sense, since they are important pieces of information indicative of utterance type (e.g., statement or question).

To summarize, the visualization results show that the three self-attention mechanisms of our model are able to adaptively focus on different information, in order to cooperatively produce a meaningful representation.

## B Baselines

### B.1 Baseline encoders

- **LD** ([Lee and Dernoncourt, 2016](#)) is a sequential sentence encoder developed for dialogue act classification. The model takes into account a fixed number of utterances from the pre-context when classifying the current one. More precisely, CNN or RNN with max-pooling is first applied separately to the current utterance and each pre-context utterance, and the resulting vectors are then aggregated through two levels of dense layers, based on two hyper-parameters,  $d_1$  and  $d_2$ , which represent the history size at level 1 and level 2 (respectively). Although the original paper reported that the CNN encoder slightly outperforms the RNN one (for DA classification), in our experiments, we used the RNN variant, since our model and the HAN baseline are RNN-based. Note that here, we used LSTM cells as [Lee and Dernoncourt \(2016\)](#) reported them to work better than GRU cells in their experiments.

- **HAN** ([Yang et al., 2016](#)). The Hierarchical Attention Network, developed for document classification, is a two-level architecture, where at level 1, each sentence in the document is separately encoded by the same sentence encoder, resulting in a sequence of sentence vectors. That sequence is then processed at level 2 by the document encoder which returns a single vector representing the entire document. The sentence and document encoders are both self-attentional bidirectional Recurrent Neural Networks (RNNs), with different parameters. We give HAN access to contextual information by feeding it the current utterance surrounded by the  $C_b$  preceding and  $C_b$  following utterances in the transcription, where  $C_b$  denotes the best context size reported in Section 7.

### B.2 Unsupervised baseline systems

- **tf-idf.** A TF-IDF vector is used as the utterance embedding, compressed to a dimension of 21 with PCA, and concatenated with the 21-dimensional discourse feature vector, thus forming a vector of dimension  $d = 42$ . This vector is then again compressed to a  $d_f = 32$ -dimensional vector. The compression steps are applied for consistency with the energy-based systems, in which textual and discourse features have the same dimensionality  $d/2 = 21$ , and the output of the utterance encoder is  $d_f$ -dimensional (see Subsection 6.3). To make

this baseline context-aware, the embeddings of the current utterance and the context utterances are averaged. In the end, FCM is applied. Note that the TF-IDF vocabulary is obtained from the entire conversation, giving this baseline a competitive advantage over the others, which never have access to the full transcription.

- **w2v.** Identical to the previous baseline, but using the average of the word2vec vectors of a given utterance instead of TF-IDF vector.
- **LCseg** is an unsupervised system adapted from previous work ([Oya et al., 2014](#); [Banerjee et al., 2015](#); [Singla et al., 2017](#)), in which disjoint topic segments are assumed to be abstractive communities. A lexical-cohesion based topic segmenter LCseg ([Galley et al., 2003](#)) is first applied on transcriptions to get the desired number of segments ( $|Q| = v/11$ ), and then only summary-worthy utterances within segments are retained for evaluation.

### B.3 Supervised baseline systems

As discussed in the literature review (see Section 2), original approaches to ACD ([Murray et al., 2012](#); [Mehdad et al., 2013](#)) are supervised and non energy-based. They have no publicly available implementations, and are hard to precisely reimplement due to lack of details about handcrafted features and dependency on external textual entailment corpora. Nevertheless, we implemented two baselines similar in spirit, taking as input the representations produced by the tf-idf and w2v unsupervised baselines previously described. More precisely, the two  $d_f$ -dimensional representations of a pair of utterances are fed into a 3-layer feed-forward neural network (with  $2d_f$ ,  $d_f$ , and 1 hidden units) which is trained on the task of predicting whether the two utterances belong to the same abstractive community or not (binary classification task). Then, like in the aforelisted studies, an utterance graph is built, where utterances are linked based on the predictions of the MLP. Finally, the CONGA algorithm ([Gregory, 2007](#)), an extension of the well-known Girvan-Newman algorithm, is applied to detect overlapping communities in the utterance graph.

## C Performance evaluation

We evaluate performance at the distance and the clustering level.

### C.1 Distance

First, we test whether the distance in the final embedding space is meaningful. To do so, for a given *query* utterance, we rank all other utterances in decreasing order of similarity with the query. We then use precision, recall, and F1 score at  $k$  to evaluate the quality of the ranking. A detailed example was provided in App. D.

Singleton communities are excluded from the evaluation at this stage. We set  $k=10$ , which is equal to the average number of non-singleton communities minus one (since the query utterance cannot be part of the results). We also report results for a variable  $k$  ( $k=v$ ), where  $k$  is equal to the size of the community of the query utterance minus one. In that case, P=R=F1.

The same procedure is repeated for all utterances. To account for differences in community size, scores are first averaged at the community-level, and then at the meeting-level. Note that the distance is Euclidean for triplet and Manhattan for siamese (see Subsections 3.2 and 3.3).

### C.2 Clustering

Second, we compare our community assignments to the human ground truth using the Omega Index (Collins and Dent, 1988), a standard metric for comparing non-disjoint clustering, used in the ACD literature (Murray et al., 2012). The Omega Index evaluates the degree of *agreement* between two clustering solutions based on *pairs* of objects being clustered. Two solutions  $s_1$  and  $s_2$  are considered to agree on a given pair of objects, if two objects are placed by both solutions in *exactly* the same *number of communities* (possibly zero). The Omega Index  $\omega$  is computed as shown in Eq. 15. The numerator is the observed agreement  $\omega_{obs}$  adjusted by expected (chance) agreement  $\omega_{exp}$ , while the denominator is the perfect agreement (value equals to 1) adjusted by expected agreement.

$$\omega(s_1, s_2) = \frac{\omega_{obs}(s_1, s_2) - \omega_{exp}(s_1, s_2)}{1 - \omega_{exp}(s_1, s_2)} \quad (15)$$

Observed and expected agreements are calculated as below:

$$\omega_{obs}(s_1, s_2) = \frac{1}{N_{total}} \sum_{j=0}^{\min(J, K)} A_j \quad (16)$$

$$\omega_{exp}(s_1, s_2) = \frac{1}{N_{total}^2} \sum_{j=0}^{\min(J, K)} N_{j1} N_{j2} \quad (17)$$

where  $A_j$  is the number of pairs agreed to be assigned to  $j$  number of communities by both solutions,  $N_{j1}$  is the number of pairs assigned to  $j$  communities in  $s_1$ ,  $N_{j2}$  is the number of pairs assigned to  $j$  communities in  $s_2$ ,  $J$  and  $K$  represent respectively the maximum number of communities in which any pair of objects appear together in solutions  $s_1$  and  $s_2$ , and  $N_{total} = n(n-1)/2$  is the total number of pairs constructed over  $n$  number of objects. To give an example, consider two clustering solutions for 5 objects:

$$s_1 = \{\{a, b, c\}, \{b, c, d\}, \{c, d, e\}, \{c, d\}\}$$

$$s_2 = \{\{a, b, c, d\}, \{b, c, d, e\}\}$$

Solutions are transformed into the table 3, from what we can obtain  $N_{total} = 10$ ,  $J = 3$ ,  $K = 2$ ,  $\min(J, K) = 2$ . Two solutions agree to place  $(a, e)$  together in no community, the pairs  $(a, b)$ ,  $(a, c)$ ,  $(c, e)$  and  $(d, e)$  in one community, and the pair  $(b, c)$  in two communities. We have  $A_0 = 1$ ,  $A_1 = 4$ ,  $A_2 = 1$ . Thus the observed agreement is  $(1 + 4 + 1)/10 = 0.6$ . Since  $N_{01} = 3$ ,  $N_{11} = 5$ ,  $N_{21} = 1$  and  $N_{02} = 1$ ,  $N_{12} = 6$ ,  $N_{22} = 3$ , the expected agreement then is  $(3 * 1 + 5 * 6 + 1 * 3)/10^2 = 0.36$ . Finally, Omega Index for this simple example is computed as:  $\omega(s_1, s_2) = (0.6 - 0.36)/(1 - 0.36) = 0.375$ .

	solution $s_1$ #communities the pair is assigned	solution $s_2$ #communities the pair is assigned	solutions $s_1$ and $s_2$ agree on the pair?
(a, b)	1	1	yes
(a, c)	1	1	yes
(a, d)	0	1	no
(a, e)	0	0	yes
(b, c)	2	2	yes
(b, d)	1	2	no
(b, e)	0	1	no
(c, d)	3	2	no
(c, e)	1	1	yes
(d, e)	1	1	yes

Table 3: Omega Index example.

Since FCM yields a probability distribution over communities for each utterance, we need to use a threshold to assign a given utterance to one or more communities. We selected 0.2 after trying multiple values in  $[0, 0.5]$  with steps of 0.05 on the validation set. Whenever one or more utterances were not assigned to any community, we merged them into a new community. Furthermore, we set the number of clusters  $|Q|$  to 11, which corresponds to the average number of ground truth communities per meeting (after merging). We also report results with a variable  $|Q|$  equal to the number of ground truth communities (variant denoted by  $|Q| = v$ ).

dist	pos	DA	role	text
0	<i>t</i>	inf	ID	<b>Um , and the rubber case requires rubber buttons , so if we definitely want plastic buttons , we shouldn't have a rubber case .</b>
0.11	-3	inf	ID	Um , we can use rubber , plastic , wood or titanium .
0.12	-5	inf	ID	<b>And al we also need a beeper or buzzer or other sort of noise thing for locating the remote .</b>
0.38	-2	sug	ID	<b>Um , I'd recommend against titanium</b>
0.42	+7	inf	ID	Um and also we should note that if we want an iPod-style wheel button , it's gonna require a m qu slightly more expensive chip .
0.54	+5	ass	ID	<b>Uh , well we can use wood .</b>
0.57	-8	inf	ID	<b>Um , standard parts include the buttons and the wheels , um the iPod-style wheel .</b>
0.68	+6	ass	ID	<b>I don't know why we'd want to .</b>
0.96	-11	inf	ID	<b>And we'll need to custom desi design a circuit board ,</b>
1.26	-13	inf	ID	<b>Um , I assume we'll be custom designing our case ,</b>
1.27	-14	inf	ID	<b>Um , so we need some custom design parts , and other parts we'll just use standard .</b>
1.43	-17	inf	ID	<b>So I've been looking at the components design .</b>
1.66	+12	off	ME	Um , can I do next ? Because I have to say something about the material
2.24	+18	inf	ME	and the findings are that the first thing to aim for is a fashion uh , fancy look and feel .
2.57	+19	inf	ME	Um . Next comes technologic technology and the innovations to do with that .
3.21	+20	inf	ME	And th last thing is the easy to use um factor .
3.92	+69	inf	UI	Uh , so people are going to be looking at this little screen .
4.02	+92	inf	ME	But the screen can come up on the telly , the she said .
...				
8.81	+623	inf	ID	It didn't give me any actual cost .
8.84	+622	inf	ID	All it said was it gave sort of relative , some chips are more expensive than others , sort of things .
8.89	+616	inf	ME	So if you throw it , it's gonna store loads of energy , and you don't need to buy a battery because they're quite f I find them annoying .
9.00	+617	sug	ME	But we need to find cost .
9.06	+621	el.inf	ME	Does anyone have costs on the on the web ?
9.95	+652	inf	PM	And you're gonna be doing protu product evaluation .
9.96	+650	inf	PM	Oh when we move on , you two are going to be playing with play-dough .
10.15	+651	inf	PM	Um , and working on the look and feel of the design and user interface design .

Table 4: Ranking example.

Note that since FCM does not return nested groupings, we merged the ground truth communities nested under the same community.

## D Ranking example

For the same utterance from the ES2011c meeting as used in App. A, we show below in Table 4 the closest and furthest utterances, in terms of Euclidean distance in the embedding space. Recall that meeting ES2011c belongs to the *validation* set. Utterances belonging to the ground truth community of the query utterance are shown in bold. Roles are ID: industrial designer, ME: marketing expert, UI: user interface designer, PM: project manager. For this example,  $P@k = v$  is equal to 77.78 (where  $v = 9$ ), and  $P$ ,  $R$ , and  $F1@k$  are 80.00, 88.89, 84.21 respectively (where  $k = 10$ ).

We can see that semantic similarity obviously

plays a role, as most of the closest utterances are about buttons and materials. But other parameters come into play. E.g., the utterances **And al we also need a beeper or buzzer or other sort of noise thing for locating the remote, and I don't know why we'd want to**, respectively ranked 2<sup>nd</sup> and 7<sup>th</sup>, are not semantically related to the query utterance. Such utterances might be placed close to the query utterance based on their positional and discourse features (speaker role and dialogue act), but also because their contexts are similar. The community where the query utterance belongs to (utterances shown in bold in the table above) is associated with the following sentence in the human abstractive summary: **The Industrial Designer gave her presentation on components and discussed which would have to be custom-made and which were standard.**

## E FCM algorithm

The goal of the Fuzzy c-Means algorithm or FCM (Bezdek et al., 1984) is to minimize the weighted within group sum of squared error objective function:

$$J(M, Q) = \sum_{q=1}^{|Q|} \sum_{t=1}^T (m_{qt})^{fuz} \|\mathbf{u}^t - \mathbf{c}_q\|_2^2 \quad (18)$$

where  $M$  and  $Q$  are the sets of membership probability distributions and community centroid vectors,  $m_{qt} \in [0, 1]$  is the probability that the  $t$ -th utterance belongs to the  $q$ -th community (with  $\sum_{q=1}^{|Q|} m_{qt} = 1$ ),  $fuz$  is a parameter that controls the amount of fuzziness,  $\|\cdot\|_2$  denotes the Euclidean distance in the triplet case (we replace it with Manhattan distance  $\|\cdot\|_1$  in the siamese case),  $\mathbf{u}^t$  is the  $t$ -th utterance vector, and  $\mathbf{c}_q$  is the  $q$ -th community centroid vector.

$M$  and  $Q$  are iteratively updated with equations:

$$m_{qt} = \left( \sum_{j=1}^{|Q|} \left( \frac{\|\mathbf{u}^t - \mathbf{c}_q\|_2}{\|\mathbf{u}^t - \mathbf{c}_j\|_2} \right)^{\frac{2}{fuz-1}} \right)^{-1} \quad (19)$$

$$\mathbf{c}_q = \frac{\sum_{t=1}^T (m_{qt})^{fuz} \mathbf{u}^t}{\sum_{t=1}^T (m_{qt})^{fuz}} \quad (20)$$

When  $fuz \rightarrow +\infty$ ,  $\forall q \in |Q|$ ,  $\forall t \in T$ ,  $m_{qt}$  tends to be equal to  $1/|Q|$ , thus utterances have identical membership to each community. While when  $fuz \rightarrow 1$ , FCM becomes equivalent to traditional  $k$ -means, in which  $m_{qt}$  is either 0 or 1 for a given utterance  $\mathbf{u}^t$  and community centroid  $\mathbf{c}_q$ . Usually in practice,  $fuz = 2$  (Schwämmle and Jensen, 2010). Learning stops when the maximum number of iterations is reached or  $J(M, Q)$  decreases by less than a predefined threshold.

## F On our choice of loss functionals

The softmax triplet loss (STL) performed better in our experiments than the margin-based loss used in (Schroff et al., 2015) and (Wang et al., 2014). One of the reasons may be that STL is able to capture a finer notion of distance. Indeed, with a margin-based loss, the Euclidean distance between the anchor and the negative (let us compactly denote it as  $d^-$ ) need to satisfy  $d^- > d^+ + m$ , where  $m$  is the margin (see Fig. 8a). In other words, the distance between the positive and the negative is at least  $m$ , when all three points are aligned.

However, the objective of STL is simply  $d^- > d^+$ , without imposing an absolute lower bound on the

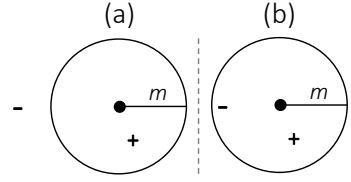


Figure 8: •, -, + denote anchor, negative, and positive.

distance between positives and negatives. That is, only the distance ratio is of interest (see Fig. 8b), which gives more freedom to the model.

For consistency, we also adopt a margin-free loss functional for siamese (MSE, see Eq. 5). It performed better than the traditional contrastive loss (Chopra et al., 2005; Neculoiu et al., 2016) in early experiments.

# Intent Detection with WikiHow

Li Zhang

Qing Lyu

Chris Callison-Burch

University of Pennsylvania

{zharry, lyuqing, ccb}@seas.upenn.edu

## Abstract

Modern task-oriented dialog systems need to reliably understand users’ intents. Intent detection is even more challenging when moving to new domains or new languages, since there is little annotated data. To address this challenge, we present a suite of pretrained intent detection models which can predict a broad range of intended goals from many actions because they are trained on wikiHow, a comprehensive instructional website. Our models achieve state-of-the-art results on the Snips dataset, the Schema-Guided Dialogue dataset, and all 3 languages of the Facebook multilingual dialog datasets. Our models also demonstrate strong zero- and few-shot performance, reaching over 75% accuracy using only 100 training examples in all datasets.<sup>1</sup>

## 1 Introduction

Task-oriented dialog systems like Apple’s Siri, Amazon Alexa, and Google Assistant have become pervasive in smartphones and smart speakers. To support a wide range of functions, dialog systems must be able to map a user’s natural language instruction onto the desired skill or API. Performing this mapping is called intent detection.

Intent detection is usually formulated as a sentence classification task. Given an utterance (e.g. “wake me up at 8”), a system needs to predict its intent (e.g. “Set an Alarm”). Most modern approaches use neural networks to jointly model intent detection and slot filling (Xu and Sarikaya, 2013; Liu and Lane, 2016; Goo et al., 2018; Zhang et al., 2019). In response to a rapidly growing range of services, more attention has been given to zero-shot intent detection (Ferreira et al., 2015a,b; Yazdani and Henderson, 2015; Chen et al., 2016; Kumar et al., 2017; Gangadharan and

Narayanaswamy, 2019). While most existing research on intent detection proposed novel model architectures, few have attempted data augmentation. One such work (Hu et al., 2009) showed that models can learn much knowledge that is important for intent detection from massive online resources such as Wikipedia.

We propose a pretraining task based on wikiHow, a comprehensive instructional website with over 110,000 professionally edited articles. Their topics span from common sense such as “How to Download Music” to more niche tasks like “How to Crochet a Teddy Bear.” We observe that the header of each step in a wikiHow article describes an action and can be approximated as an utterance, while the title describes a goal and can be seen as an intent. For example, “find good gas prices” in the article “How to Save Money on Gas” is similar to the utterance “where can I find cheap gas?” with the intent “Save Money on Gas.” Hence, we introduce a dataset based on wikiHow, where a model predicts the goal of an action given some candidates. Although most of wikiHow’s domains are far beyond the scope of any present dialog system, models pretrained on our dataset would be robust to emerging services and scenarios. Also, as wikiHow is available in 18 languages, our pretraining task can be readily extended to multilingual settings.

Using our pretraining task, we fine-tune transformer language models, achieving state-of-the-art results on the intent detection task of the Snips dataset (Coucke et al., 2018), the Schema-Guided Dialog (SGD) dataset (Rastogi et al., 2019), and all 3 languages (English, Spanish, and Thai) of the Facebook multilingual dialog datasets (Schuster et al., 2019), with statistically significant improvements. As our accuracy is close to 100% on all these datasets, we further experiment with zero- or few-shot settings. Our models achieve over 70% accuracy with no in-domain training data on Snips

<sup>1</sup>The data and models are available at <https://github.com/zharry29/wikihow-intent>.

and SGD, and over 75% with only 100 training examples on all datasets. This highlights our models’ ability to quickly adapt to new utterances and intents in unseen domains.

## 2 WikiHow Pretraining Task

### 2.1 Corpus

We crawl the wikiHow website in English, Spanish, and Thai (the languages were chosen to match those in the Facebook multilingual dialog datasets). We define the **goal** of each article as its title stripped of the prefix “How to” (and its equivalent in other languages). We extract a set of **steps** for each article, by taking the bolded header of each paragraph.

### 2.2 WikiHow Pretraining Dataset

A wikiHow article’s goal can approximate an intent, and each step in it can approximate an associated utterance. We formulate the pretraining task as a 4-choose-1 multiple choice format: given a step, the model infers the correct goal among 4 candidates. For example, given the step “let check-in agents and flight attendants know if it’s a special occasion” and the candidate goals:

- A. Get Upgraded to Business Class
- B. Change a Flight Reservation
- C. Check Flight Reservations
- D. Use a Discount Airline Broker

the correct goal would be A. This is similar to intent detection, where a system is given a user utterance and then must select a supported intent.

We create intent detection pretraining data using goal-step pairs from each wikiHow article. Each article contributes at least one positive goal-step pair. However, it is challenging to sample negative candidate goals for a given step. There are two reasons for this. First, random sampling of goals correctly results in true negatives, but they tend to be so distant from the positive goal that the classification task becomes trivial and the model does not learn sufficiently. Second, if we sample goals that are similar to the positive goal, then they might not be true negatives, since there are many steps in wikiHow often with overlapping goals. To sample high-quality negative training instances, we start with the correct goal and search in its article’s “related articles” section for an article whose title has the least lexical overlap with the current goal. We recursively do this until we have enough candidates. Empirically, examples created this way are mostly clean, with an example shown above. We select one

positive goal-step pair from each article by picking its longest step. In total, our wikiHow pretraining datasets have 107,298 English examples, 64,803 Spanish examples, and 6,342 Thai examples.

## 3 Experiments

We fine-tune a suite of off-the-shelf language models pretrained on our wikiHow data, and evaluate them on 3 major intent detection benchmarks.

### 3.1 Models

We fine-tune a pretrained RoBERTa model (Liu et al., 2019) for the English datasets and a pretrained XLM-RoBERTa model (Conneau et al., 2019) for the multilingual datasets. We cast the instances of the intent detection datasets into a multiple-choice format, where the utterance is the input and the full set of intents are the possible candidates, consistent with our wikiHow pretraining task. For each model, we append a linear classification layer with cross-entropy loss to calculate a likelihood for each candidate, and output the candidate with the maximum likelihood.

For each intent detection dataset in any language, we consider the following settings:

**+in-domain (+ID)**: a model is only trained on the dataset’s in-domain training data;

**+wikiHow +in-domain (+WH+ID)**: a model is first trained on our wikiHow data in the corresponding language, and then trained on the dataset’s in-domain training data;

**+wikiHow zero-shot (+WH 0-shot)**: a model is trained only on our wikiHow data in the corresponding language, and then applied directly to the dataset’s evaluation data.

For non-English languages, the corresponding wikiHow data might suffer from smaller sizes and lower quality. Hence, we additionally consider the following cross-lingual transfer settings for non-English datasets:

**+en wikiHow +in-domain (+enWH+ID)**, a model is trained on wikiHow data in English, before it is trained on the dataset’s in-domain training data;

**+en wikiHow zero-shot (+enWH 0-shot)**, a model is trained on wikiHow data in English, before it is directly applied to the dataset’s evaluation data.

### 3.2 Datasets

We consider the 3 following benchmarks:

**The Snips dataset** (Coucke et al., 2018) is a single-turn English dataset. It is one of the most cited dialog benchmarks in recent years, containing

	Training Size	Valid. Size	Test Size	Num. Intents
Snips	2,100	700	N/A	7
SGD	163,197	24,320	42,922	4
FB-en	30,521	4,181	8,621	12
FB-es	3,617	1,983	3,043	12
FB-th	2,156	1,235	1,692	12

Table 1: Statistics of the dialog benchmark datasets.

utterances collected from the Snips personal voice assistant. While its full training data has 13,784 examples, we find that our models only need its smaller training split consisting of 2,100 examples to achieve high performance. Since Snips does not provide test sets, we use the validation set for testing and the full training set for validation. Snips involves 7 intents, including *Add to Playlist*, *Rate Book*, *Book Restaurant*, *Get Weather*, *Play Music*, *Search Creative Work*, and *Search Screening Event*. Some example utterances include “Play the newest melody on Last Fm by Eddie Vinson,” “Find the movie schedule in the area,” etc.

**The Schema-Guided Dialogue dataset** (SGD) ([Rastogi et al., 2019](#)) is a multi-turn English dataset. It is the largest dialog corpus to date spanning dozens of domains and services, used in the DSTC8 challenge ([Rastogi et al., 2020](#)) with dozens of team submissions. Schemas are provided with at most 4 intents per dialog turn. Examples of these intents include *Buy Movie Tickets for a Particular show*, *Make a Reservation with the Therapist*, *Book an Appointment at a Hair Stylist*, *Browse attractions in a given city*, etc. At each turn, we use the last 3 utterances as input. An example: “That sounds fun. What other attractions do you recommend? There is a famous place of worship called Akshardham.”

**The Facebook multilingual datasets** (FB-en/es/th) ([Schuster et al., 2019](#)) is a single-turn multilingual dataset. It is the only multilingual dialog dataset to the best of our knowledge, containing utterances annotated with intents and slots in English (en), Spanish (es), and Thai (th). It involves 12 intents, including *Set Reminder*, *Check Sunrise*, *Show Alarms*, *Check Sunset*, *Cancel Reminder*, *Show Reminders*, *Check Time Left on Alarm*, *Modify Alarm*, *Cancel Alarm*, *Find Weather*, *Set Alarm*, and *Snooze Alarm*. Some example utterances are “Is my alarm set for 10 am today?” “Colocar una alarma para mañana a las 3 am,” ฉันมีนาฬิกาปลุกสำหรับตอนเช้ามื้ย etc.

	Snips	SGD	FB-en
(Ren and Xue, 2020)	.993	N/A	.993
(Ma et al., 2019)	N/A	.948	N/A
+in-domain (+ID)	.990	.942	.993
(ours) +WH+ID	<b>.994</b>	<b>.951</b> †	<b>.995</b> †
(ours) +WH 0-shot	.713	.787	.445
Chance	.143	.250	.083

Table 2: The accuracy of intent detection on English datasets using RoBERTa. State-of-the-art performances are in bold; † indicates statistically significant improvement from the previous state-of-the-art.

	FB-en	FB-es	FB-th
(Ren and Xue, 2020)	.993	N/A	N/A
(Zhang et al., 2019)	N/A	.978	.967
+in-domain (+ID)	.993	.986	.962
(ours) +WH+ID	<b>.995</b>	.988	.971
(ours) +enWH+ID	<b>.995</b>	<b>.990</b> †	<b>.976</b> †
(ours) +WH 0-shot	.416	.129	.119
(ours) +enWH 0-shot	.416	.288	.124
Chance	.083	.083	.083

Table 3: The accuracy of intent detection on multilingual datasets using XLM-RoBERTa.

Statistics of the datasets are shown in Table 1.

### 3.3 Baselines

We compare our models with the previous state-of-the-art results of each dataset:

- [Ren and Xue \(2020\)](#) proposed a Siamese neural network with triplet loss, achieving state-of-the-art results on Snips and FB-en;
- [Zhang et al. \(2019\)](#) used multi-task learning to jointly learn intent detection and slot filling, achieving state-of-the-art results on FB-es and FB-th;
- [Ma et al. \(2019\)](#) augmented the data via back-translation to and from Chinese, achieving state-of-the-art results on SGD.

### 3.4 Modelling Details

After experimenting with base and large models, we use RoBERTa-large for the English datasets and XLM-RoBERTa-base for the multilingual dataset for best performances. All our models are implemented using the HuggingFace Transformer library<sup>2</sup>.

We tune our model hyperparameters on the validation sets of the datasets we experiment with. However, in all cases, we use a unified setting

<sup>2</sup><https://github.com/huggingface/transformers>

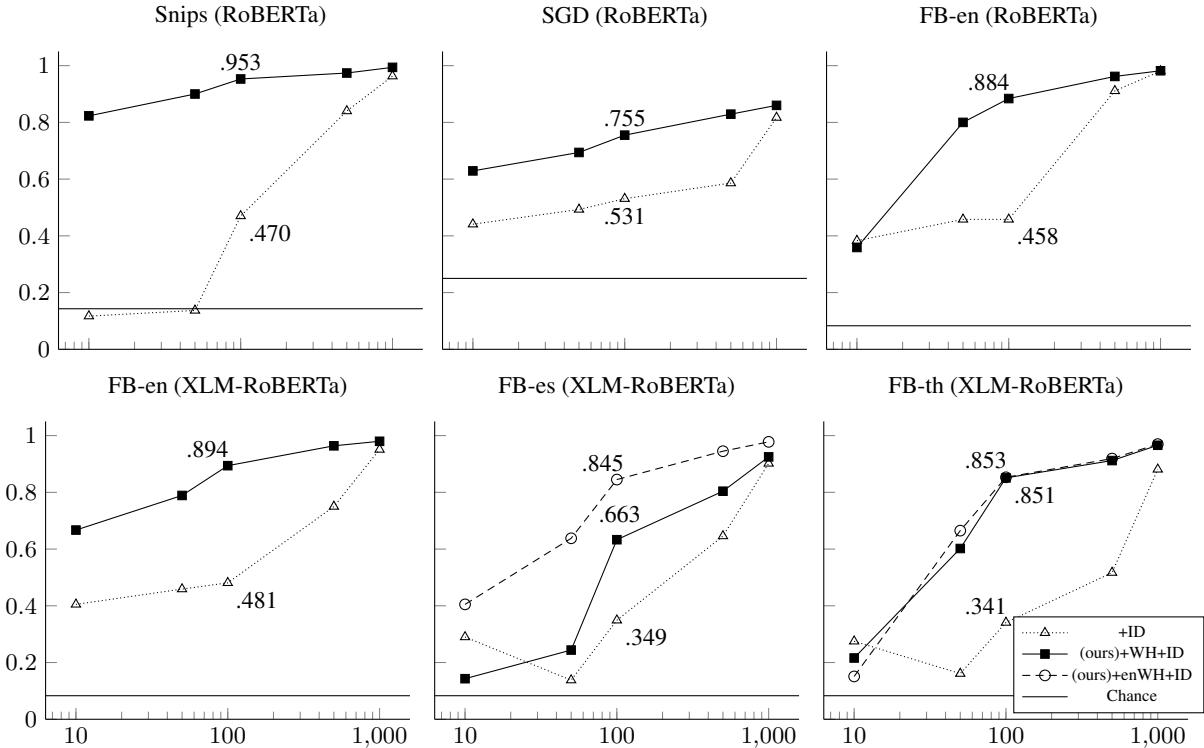


Figure 1: Learning curves of models in low-resource settings. The vertical axis is the accuracy of intent detection, while the horizontal axis is the number of in-domain training examples of each task, distorted to log-scale.

which empirically performs well, using the Adam optimizer (Kingma and Ba, 2014) with an epsilon of  $1e^{-8}$ , a learning rate of  $5e^{-6}$ , maximum sequence length of 80 and 3 epochs. We variate the batch size from 2 to 16 according to the number of candidates in the multiple-choice task, to avoid running out of memory. We save the model every 1,000 training steps, and choose the model with the highest validation performance to be evaluated on the test set.

We run our experiments on an NVIDIA GeForce RTX 2080 Ti GPU, with half-precision floating point format (FP16) with O1 optimization. Each epoch takes up to 90 minutes in the most resource intensive setting, i.e. running a RoBERTa-large on around 100,000 training examples of our wikiHow pretraining dataset.

### 3.5 Results

The performance of RoBERTa on the English datasets (Snips, SGD, and FB-en) are shown in Table 2. We repeat each experiment 20 times, report the mean accuracy, and calculate its p-value against the previous state-of-the-art result, using a one-sample and one-tailed t-test with a significance level of 0.05. Our models achieve state-of-the-art results using the available in-domain training data. Moreover, our wikiHow data enables our models to

demonstrate strong performances in zero-shot settings with no in-domain training data, implying our models’ strong potential to adapt to new domains.

The performance of XLM-RoBERTa on the multilingual datasets (FB-en, FB-es, and FB-th) are shown in Table 3. Our models achieve state-of-the-art results on all 3 languages. While our wikiHow data in Spanish and Thai does improve models’ performances, its effect is less salient than the English wikiHow data.

Our experiments above focus on settings where all available in-domain training data are used. However, modern task-oriented dialog systems must rapidly adapt to burgeoning services (e.g. Alexa Skills) in different languages, where little training data are available. To simulate low-resource settings, we repeat the experiments with exponentially increasing number of training examples up to 1,000. We consider the models trained only on in-domain data (+ID), those first pretrained on our wikiHow data in corresponding languages (+WH+ID), and those first pretrained on our English wikiHow data (+enWH+ID) for FB-es and FB-th.

The learning curves of each dataset are shown in Figure 1. Though the vanilla transformers models (+ID) achieve close to state-of-the-art performance with access to the full training data (see Table 2

and 3), they struggle in the low-resource settings. When given up to 100 in-domain training examples, their accuracies are less than 50% on most datasets. In contrast, our models pretrained on our wikiHow data (+WH+ID) can reach over 75% accuracy given only 100 training examples on all datasets.

## 4 Discussion and Future Work

As our model performances exceed 99% on Snips and FB-en, the concern arises that these intent detection datasets are “solved”. We address this by performing error analysis and providing future outlooks for intent detection.

### 4.1 Error Analysis

Our model misclassifies 7 instances in the Snips test set. Among them, 6 utterances include proper nouns on which intent classification is contingent. For example, the utterance “please open Zvooq” assumes the knowledge that Zvooq is a streaming service, and its labelled intent is “Play Music.”

Our model misclassifies 43 instances in the FB-en test set. Among them, 10 has incorrect labels: e.g. the labelled intent of “have alarm go off at 5 pm” is “Show Alarms,” while our model prediction “Set Alarm” is in fact correct. 28 are ambiguous: e.g. the labelled intent of “repeat alarm every weekday” is “Set Alarm,” whereas that of “add an alarm for 2:45 on every Monday” is “Modify Alarm.” We only find 1 example an interesting edge case: the gold intent of “remind me if there will be a rain forecast tomorrow” is “Find Weather,” while our model incorrectly chooses “Set Reminder.”

By performing manual error analyses on our model predictions, we observe that most misclassified examples involve ambiguous wordings, wrong labels, or obscure proper nouns. Our observations imply that Snips and FB-en might be too easy to effectively evaluate future models.

### 4.2 Open-Domain Intent Detection

State-of-the-art models now achieve greater than 99% percent accuracy on standard benchmarks for intent detection. However, intent detection is far from being solved. The standard benchmarks only have a dozen intents, but future dialog systems will need to support many more functions with intents from a wide range of domains. To demonstrate that our pretrained models can adapt to unseen, open-domain intents, we hold out 5,000 steps (as utterances) with their corresponding goals (as intents) from our wikiHow dataset as a proxy of an

intent detection dataset with more than 100,000 possible intents (all goals in wikiHow).

For each step, we sample 100 goals with the highest embedding similarity to the correct goal, as most other goals are irrelevant. We then rank them for the likelihood that the step helps achieve them. Our RoBERTa model achieves a mean reciprocal rank of 0.462 and a 36% accuracy of ranking the correct goal first. As a qualitative example, given the step “find the order that you want to cancel,” the top 3 ranked steps are “Cancel an Order on eBay”, “Cancel an Online Order”, “Cancel an Order on Amazon.” This hints that our pretrained models’ can work with a much wider range of intents than those in current benchmarks, and suggests that future intent detection research should focus on open domains, especially those with little data.

## 5 Conclusion

By pretraining language models on wikiHow, we attain state-of-the-art results in 5 major intent detection datasets spanning 3 languages. The wide-ranging domains and languages of our pretraining resource enable our models to excel with few labelled examples in multilingual settings, and suggest open-domain intent detection is now feasible.

## Acknowledgments

This research is based upon work supported in part by the DARPA KAIROS Program (contract FA8750-19-2-1004), the DARPA LwLL Program (contract FA8750-19-2-0201), and the IARPA BETTER Program (contract 2019-19051600004). Approved for Public Release, Distribution Unlimited. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of DARPA, IARPA, or the U.S. Government.

We thank the anonymous reviewers for their valuable feedback.

## References

- Yun-Nung Chen, Dilek Hakkani-Tür, and Xiaodong He. 2016. Zero-shot learning of intent embeddings for expansion by convolutional deep structured semantic models. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6045–6049. IEEE.
- Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco

- Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.
- Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*.
- Emmanuel Ferreira, Bassam Jabaian, and Fabrice Lefèvre. 2015a. Zero-shot semantic parser for spoken language understanding. In *Sixteenth Annual Conference of the International Speech Communication Association*.
- Emmanuel Ferreira, Bassam Jabaian, and Fabrice Lefèvre. 2015b. Online adaptative zero-shot learning spoken language understanding using word-embedding.
- Rashmi Gangadharaiyah and Balakrishnan Narayanaswamy. 2019. Joint multiple intent detection and slot labeling for goal-oriented dialog. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 564–569, Minneapolis, Minnesota. Association for Computational Linguistics.
- Chih-Wen Goo, Guang Gao, Yun-Kai Hsu, Chih-Li Huo, Tsung-Chieh Chen, Keng-Wei Hsu, and Yun-Nung Chen. 2018. Slot-gated modeling for joint slot filling and intent prediction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 753–757, New Orleans, Louisiana. Association for Computational Linguistics.
- Jian Hu, Gang Wang, Fred Lochofsky, Jian-tao Sun, and Zheng Chen. 2009. Understanding user’s query intent with wikipedia. In *Proceedings of the 18th International Conference on World Wide Web, WWW ’09*, page 471–480, New York, NY, USA. Association for Computing Machinery.
- Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Anjishnu Kumar, Pavankumar Reddy Muddireddy, Markus Dreyer, and Björn Hoffmeister. 2017. Zero-shot learning across heterogeneous overlapping domains.
- Bing Liu and Ian Lane. 2016. Attention-based recurrent neural network models for joint intent detection and slot filling.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.
- Yue Ma, Zengfeng Zeng, Dawei Zhu, Xuan Li, Yiyi Yang, Xiaoyuan Yao, Kaijie Zhou, and Jianping Shen. 2019. An end-to-end dialogue state tracking system with machine reading comprehension and wide & deep classification.
- Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghu Gupta, and Pranav Khaitan. 2019. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. *arXiv preprint arXiv:1909.05855*.
- Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghu Gupta, and Pranav Khaitan. 2020. Schema-guided dialogue state tracking task at dstc8.
- F. Ren and S. Xue. 2020. Intention detection based on siamese neural network with triplet loss. *IEEE Access*, 8:82242–82254.
- Sebastian Schuster, Sonal Gupta, Rushin Shah, and Mike Lewis. 2019. Cross-lingual transfer learning for multilingual task oriented dialog. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3795–3805, Minneapolis, Minnesota. Association for Computational Linguistics.
- P. Xu and R. Sarikaya. 2013. Convolutional neural network based triangular crf for joint intent detection and slot filling. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 78–83.
- Majid Yazdani and James Henderson. 2015. A model of zero-shot learning of spoken language understanding. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 244–249, Lisbon, Portugal. Association for Computational Linguistics.
- Chenwei Zhang, Yaliang Li, Nan Du, Wei Fan, and Philip Yu. 2019. Joint slot filling and intent detection via capsule neural networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5259–5267, Florence, Italy. Association for Computational Linguistics.
- Z. Zhang, Z. Zhang, H. Chen, and Z. Zhang. 2019. A joint learning framework with bert for spoken language understanding. *IEEE Access*, 7:168849–168858.

# A Systematic Characterization of Sampling Algorithms for Open-ended Language Generation

Moin Nadeem\*

Massachusetts Institute of Technology  
mnaadeem@mit.edu

Kyunghyun Cho

New York University  
kyunghyun.cho@nyu.edu

Tianxing He\*

Massachusetts Institute of Technology  
cloudygoose@csail.mit.edu

James Glass

Massachusetts Institute of Technology  
glass@mit.edu

## Abstract

This work studies the widely adopted ancestral sampling algorithms for auto-regressive language models, which is not widely studied in the literature. We use the quality-diversity (Q-D) trade-off to investigate three popular sampling algorithms (top- $k$ , nucleus and tempered sampling). We focus on the task of open-ended language generation. We first show that the existing sampling algorithms have similar performance. After carefully inspecting the transformations defined by different sampling algorithms, we identify three key properties that are shared among them: *entropy reduction*, *order preservation*, and *slope preservation*. To validate the importance of the identified properties, we design two sets of new sampling algorithms: one set in which each algorithm satisfies all three properties, and one set in which each algorithm violates at least one of the properties. We compare their performance with existing sampling algorithms, and find that violating the identified properties could lead to drastic performance degradation, as measured by the Q-D trade-off. On the other hand, we find that the set of sampling algorithms that satisfies these properties performs on par with the existing sampling algorithms.<sup>1</sup>

## 1 Introduction

A language model (LM) is a central module for natural language generation (NLG) tasks (Young et al., 2018) such as machine translation (Wu et al., 2018), dialogue response generation (Li et al., 2017), image captioning (Lin et al.), and related tasks. Given a trained LM, finding the best way to generate a sample from it has been an important challenge for NLG applications.

\*Equal contribution.

<sup>1</sup>Our data and code are available at <https://github.com/moinnadeem/characterizing-sampling-algorithms>.

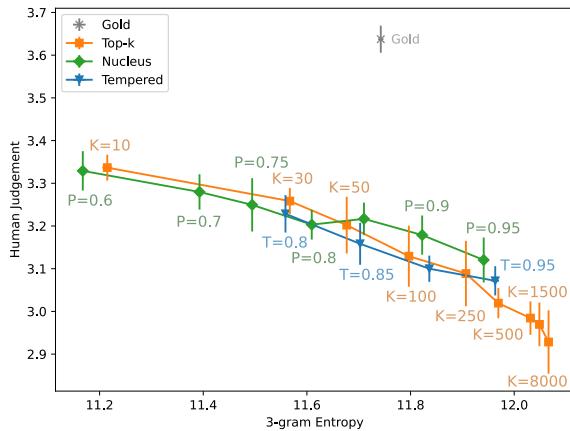


Figure 1: Human evaluation (y-axis: quality, x-axis: diversity, both are the bigger the better) shows that the generation performance of existing sampling algorithms are on par with each other.

Decoding, i.e., finding the most probable output sequence from a trained model, is a natural principle for generation. The beam-search decoding algorithm approximately finds the most likely sequence by performing breadth-first search over a restricted search space. It has achieved success in machine translation, summarization, image captioning, and other subfields.

However, in the task of open-ended language generation (which is the focus of this work), a significant degree of *diversity* is required. For example, conditioned on the prompt “The news says that . . .”, the LM is expected to be able to generate a wide range of interesting continuations. While the deterministic behavior of decoding algorithms could give high-quality samples, they suffer from a serious lack of diversity.

This need for diversity gives rise to a wide adoption of various sampling algorithms. Notably, top- $k$  sampling (Fan et al., 2018), nucleus sampling (Holtzman et al., 2020), and tempered sampling (Caccia et al., 2020) have been used in open-ended

generation (Radford et al., 2018; Caccia et al., 2020), story generation (Fan et al., 2018), and dialogue response generation (Zhang et al., 2020b). However, the sampling algorithm and the hyperparameter are usually chosen via heuristics, and a comprehensive comparison between existing sampling algorithm is lacking in the literature. More importantly, **the underlying reasons behind the success of the existing sampling algorithms still remains poorly understood.**

In this work, we begin by using the quality-diversity (Q-D) trade-off (Caccia et al., 2020) to compare the three existing sampling algorithms. For automatic metrics, we use the BLEU score for quality and n-gram entropy for diversity. We also correlate these automatic metrics with human judgements. The first observation we draw is that top- $k$ , nucleus and tempered sampling perform on par in the Q-D trade-off, as shown in Figure 1. Motivated by this result, we extract three key properties by inspecting the transformations defined by the sampling algorithms: (1) *entropy reduction*, (2) *order preservation* and (3) *slope preservation*. We prove all three properties hold for the three existing sampling algorithms.

We then set out to systematically validate the importance of the identified properties. To do so, we design two sets of new sampling algorithms in which each algorithm either violates one of the identified properties, or satisfies all properties. Using the Q-D trade-off, we compare their efficacy against existing algorithms, and find that violating these identified properties could result in significant performance degradation. More interestingly, we find that the set of sampling algorithms that satisfies these properties has generation performance that matches the performance of existing sampling algorithms.

## 2 Sampling Algorithms for Autoregressive Language Models

### 2.1 Autoregressive Language Modeling

The task of autoregressive language modeling is to learn the probability distribution of the  $(l+1)$ -th word  $W_{l+1}$  in a sentence  $W$  conditioned on the word history  $W_{1:l} := (W_1, \dots, W_l)$  and context  $C$ . Here, we use  $W_i \in V$  to denote a discrete random variable distributed across a fixed vocabulary  $V$ . In this work, the vocabulary is constructed on subword level (Sennrich et al., 2016).

Given a training set  $D$ , maximum likelihood es-

timation (MLE) has been the most popular framework to train an autoregressive LM (Mikolov et al., 2010). MLE training minimizes the negative log-likelihood (NLL) objective below:

$$L_{\text{MLE}} = \frac{1}{|D|} \sum_{(W, C) \in D} -\sum_{l=0}^{L-1} \log P_\theta(W_{l+1}|W_{1:l}, C), \quad (1)$$

where  $\theta$  denotes model parameters, and  $P_\theta(\cdot | W_{1:l})$  denotes the conditional model distribution of  $W_{l+1}$  given a prefix  $W_{1:l}$ . For simplicity, we assume all sentences are of length  $L$  in the formulations. Since this work focuses on sampling from a given model instead of training it, in the rest of the paper, we abbreviate  $P_\theta(\cdot)$  as  $P(\cdot)$  for brevity.

### 2.2 Existing Sampling Algorithms

Given a trained LM and a context  $C$ , an ancestral sampling algorithm seeks to generate a sequence from  $P(W|C)$  by sampling token-by-token from a transformed version of  $P(W_{l+1}|W_{1:l}, C)$ . We now review and formulate three popular sampling algorithms: top- $k$  (Fan et al., 2018), nucleus (Holtzman et al., 2020), and tempered (Ackley et al., 1985; Caccia et al., 2020) sampling.

We view these algorithms as different transformations applied to the distribution  $P(W_{l+1}|W_{1:l}, C)$ . First, we treat the conditional distribution  $P(W_{l+1}|W_{1:l}, C)$  as a *sorted* vector  $p$  of length  $|V|$ . By sorting, we rearrange the elements such that if  $i < j \rightarrow p_i >= p_j$ .<sup>2</sup> We list the transformations and their intuition below:

**Definition 2.1. (Top- $k$ )** In top- $k$  sampling, we only sample from the top  $K$  tokens:

$$\hat{p}_i = \frac{p_i \cdot \mathbb{1}\{i \leq K\}}{\sum_{j=1}^K p_j}, \quad (2)$$

where  $\mathbb{1}$  is the indicator function, and  $K$  ( $1 \leq K \leq |V|$ ) is the hyperparameter.

**Definition 2.2. (Nucleus)** With a hyperparameter  $P$  ( $0 < P \leq 1$ ), in nucleus sampling, we sample from the top- $P$  mass of  $p$ :

$$\hat{p}_i = \frac{p'_i}{\sum_{j=1}^{|V|} p'_j}, \quad (3)$$

where  $p'_i = p_i \cdot \mathbb{1}\{\sum_{j=1}^{i-1} p_j < P\}$ .

<sup>2</sup>The token indexes are also permuted accordingly.

**Definition 2.3. (Tempered)** In tempered sampling, the log probabilities are scaled by  $\frac{1}{T}$ :

$$\hat{p}_i = \frac{\exp(\log(p_i)/T)}{\sum_{j=1}^{|V|} \exp(\log(p_j)/T)}. \quad (4)$$

In this work, we assume  $0 < T < 1$ , i.e., the distribution is only made sharper<sup>3</sup>.

We additionally experiment with a combined version of top- $k$  and tempered sampling:

**Definition 2.4. (Tempered Top- $k$ )** We combine the transformation defined by top- $k$  and tempered sampling:

$$\hat{p}_i = \frac{p'_i}{\sum_{j=1}^{|V|} p'_j}, \quad (5)$$

where  $p'_i = \exp(\log(p_i)/T) \cdot \mathbb{1}\{i \leq K\}$ . We set  $1 \leq K \leq |V|$  and  $0 < T < 1$ .

Throughout this work we use  $\hat{p}$  to denote the normalized version of the transformed distribution. All algorithms have hyperparameters to control the entropy of the transformed distribution. For example,  $K$  in top- $k$  sampling controls the size of the support of the resulting distribution. We will formalize this statement in Property 1 below.

### 3 Properties of Sampling Algorithms

As we will show in Section 5.1 (also Figure 1), top- $k$ , nucleus and tempered sampling perform on par with each other under our evaluation. This key observation makes us question: *What are the core principles underlying the different algorithms that lead to their similar performance?*

To answer this question, in this section, we identify three core properties that are provably shared by the existing sampling algorithms. We then design experiments to validate their importance.

#### 3.1 Identifying Core Properties

By inspecting the transformations listed in Definition 2.1, 2.2 and 2.3, we extract the following three properties:

**Property 1. (Entropy Reduction)**: The transformation strictly decrease the entropy of the distribution. Formally,  $\mathcal{H}(\hat{p}) < \mathcal{H}(p)$ , where  $\mathcal{H}(p) = -\sum_{i=1}^{|V|} p_i \log p_i$ .

---

<sup>3</sup>One could also use  $T > 1$ , but it does not work well in practice.

**Property 2. (Order Preservation)**: The order of the elements in the distribution is preserved. Formally,  $p_i \geq p_j \rightarrow \hat{p}_i \geq \hat{p}_j$ .

**Property 3. (Slope Preservation)**: The “slope” of the distribution is preserved. Formally,  $\forall \hat{p}_i > \hat{p}_j > \hat{p}_k > 0$  (i.e., they are not truncated), we have  $\frac{\log p_i - \log p_j}{\log p_j - \log p_k} = \frac{\log \hat{p}_i - \log \hat{p}_j}{\log \hat{p}_j - \log \hat{p}_k}$ .

The order preservation property implies that truncation can only happen in the tail of the distribution, which aligns with top- $k$  and nucleus sampling. The slope preservation property is stronger than the order preservation property in that not only the ordering, but also the relative magnitude of the elements in the distribution needs to be somewhat preserved by the transformation.

All these three properties are shared by the three existing sampling algorithms:

**Proposition 1.** Property 1, 2 and 3 hold for the top- $k$ , nucleus and tempered sampling transformations formulated in Definitions 2.1, 2.2 and 2.3.

*Proof.* See Appendix B.  $\square$

We then set out to validate the importance of these identified properties in the aspects of *necessity* and *sufficiency*. To do so, we design two sets of new sampling algorithms in which each algorithm either violates one of the identified properties, or satisfies all properties. We list them in the next section.

#### 3.2 Designed Sampling Algorithms

**Property-violating algorithms** To validate the necessity of each property, we design several sampling algorithms which *violate at least one of the identified properties*. In our experiments, we check whether that violation leads to a significant degradation in performance. We list them below:

**Definition 3.1. (Target Entropy)** Based on tempered sampling, target entropy sampling tunes the temperature  $t$  such that the transformed distribution has entropy value equal to the hyperparameter  $E$  ( $0 < E \leq \log |V|$ ). We formulate it below:

$$\hat{p}_i = \frac{\exp(\log(p_i)/t)}{\sum_{j=1}^{|V|} \exp(\log(p_j)/t)}, \quad (6)$$

where  $t$  is selected such that  $H(\hat{p}) = E$ .

Target entropy sampling violates entropy reduction, because when  $H(p) < E$ , the entropy will be tuned up (i.e.,  $H(\hat{p}) > H(p)$ ).

**Definition 3.2. (Random Mask)** In random mask sampling, we randomly mask out tokens in the distribution with rate  $R$ . We formulate it below:

$$\hat{p}_i = \frac{p'_i}{\sum_{j=1}^{|V|} p'_j}, \quad (7)$$

where  $p'_i = p_i \cdot \mathbb{1}\{i = 1 \text{ or } u_i > R\}$  and  $u_i \sim U(0, 1)$ . The hyperparameter  $R$  ( $0 < R \leq 1$ ) controls the size of the support of the resulting distribution. In Appendix A, we show it is crucial that the token which is assigned the largest probability ( $p_1$ ) is never be masked.

Random mask sampling is different from top- $k$  or nucleus sampling in that the masking not only happens in the tail of the distribution. Therefore, it violates the order preservation property.

**Definition 3.3. (Noised Top- $k$ )** We add a *sorted* noise distribution to the result from top- $K$  transformation, and the weight of the noise distribution is controlled by a hyperparameter  $W$  ( $0 \leq W \leq 1$ ). We formulate it below:

$$\hat{p} = (1 - W)\hat{p}^{\text{top-K}} + Wp^{\text{noise-K}}, \quad (8)$$

where  $p^{\text{noise-K}}$  is a uniformly sampled *sorted K-simplex*, which satisfies  $\sum_{i=1}^K p_i^{\text{noise-K}} = 1$  and  $i < j \rightarrow p_i^{\text{noise-K}} \geq p_j^{\text{noise-K}} \geq 0$ .

The sorted nature of the noise distribution  $p^{\text{noise-K}}$  maintains order preservation. However, it violates slope preservation, and the noise weight  $W$  controls the degree of the violation.

**Property-satisfying algorithms** To validate the sufficiency of the identified properties, we design two new sampling algorithms for which *all three properties hold*. And in our experiments we check whether their performance is on par with the existing sampling algorithms. We list them below:

**Definition 3.4. (Random Top- $k$ )** We design a randomized version of top- $k$  sampling: At each time step, we sample a uniformly random float number  $u \sim U(0, 1)$ , and use it to specify a top- $k$  truncation:

$$\hat{p}_i = \frac{p_i \cdot \mathbb{1}\{i \leq k\}}{\sum_{j=1}^k p_j}, \quad (9)$$

where  $k = \lfloor 1 + M \cdot u \rfloor$ . The hyperparameter  $M$  ( $1 \leq M < |V|$ ) controls the maximum truncation threshold.

**Definition 3.5. (Max Entropy)** Max entropy sampling is similar to target entropy sampling (Definition 3.1). However to match entropy reduction (Property 1), we only tune the temperature when  $\mathcal{H}(p) > E$ , where  $E$  is the hyperparameter ( $0 < E \leq \log |V|$ ):

$$\hat{p}_i = \begin{cases} \frac{\exp(\log(p_i)/t)}{\sum_{j=1}^{|V|} \exp(\log(p_j)/t)}, & \text{if } \mathcal{H}(p) > E \\ p_i, & \text{otherwise} \end{cases}, \quad (10)$$

where  $t$  is selected so that  $\mathcal{H}(\hat{p}) = E$ .

It is easy to prove that Property 1, 2, and 3 holds for the transformations defined by random top- $k$  and max entropy sampling, and we omit the proof for brevity.

## 4 Experiment Setup

In this section, we first establish evaluation protocols, and then describe the model and data we use for the open-ended language generation task.

### 4.1 Evaluation via the Q-D Trade-off

How to efficiently measure the generation performance of a NLG model has been an important open question. Most existing metrics either measure the *quality* aspect (e.g. BLEU score) or the *diversity* (e.g. n-gram entropy) aspect. To make the situation more complicated, each sampling algorithm has its own hyperparameters which controls the trade-off between quality and diversity.

To address the challenges above, we adopt the quality-diversity trade-off proposed by Caccia et al. (2020). In the Q-D trade-off, we perform a fine-grained sweep of hyperparameters for each sampling algorithm, and compute the quality and diversity score for each configuration. We report two pairs of Q/D metrics, with one pair using automatic evaluation and the other using human evaluation. In the next two sections, we describe the metrics we use, and refer readers to Caccia et al. (2020) for more intuition behind the Q-D trade-off.

#### 4.1.1 Automatic Evaluation

For automatic metrics, we adopt the corpus-BLEU (Yu et al., 2016) metric to measure quality and the self-BLEU (Zhu et al., 2018) metric to measure diversity. We formulate them below.

Given a batch of generated sentences  $S_{\text{gen}}$  and a batch of sentences from ground-truth data as references  $S_{\text{ref}}$ , corpus-BLEU returns the average

BLEU score (Papineni et al., 2002) of every model generated sentence against the reference set:

$$\text{corpus-BLEU}(S_{\text{gen}}, S_{\text{ref}}) = \frac{1}{|S_{\text{gen}}|} \sum_{W \in S_{\text{gen}}} \text{BLEU}(W, S_{\text{ref}}). \quad (11)$$

A higher corpus-BLEU score means that the generated sequences has better quality in that it has higher ngram-level overlap with the reference data. Based on the same intuition, we define the self-BLEU metric to quantify the diversity aspect:

$$\text{self-BLEU}(S_{\text{gen}}) = \text{corpus-BLEU}(S_{\text{gen}}, S_{\text{gen}}), \quad (12)$$

where a lower self-BLEU score means that the samples have better diversity.

In our experiments, we feed the first ten subwords of every sample from test set to the model, and compare the model-generated sequences to the reference samples in the validation set. We use 10,000 samples to compute corpus-BLEU or self-BLEU, i.e.,  $|S_{\text{gen}}| = |S_{\text{ref}}| = 10,000$ .

Automatic evaluation enables us to do a fine-grained sweep of the hyperparameters for each sampling algorithm, and compare them in the quality-diversity trade-off. However, observations from automatic evaluation could be misaligned with human evaluation (Belz and Reiter, 2006). Therefore, we confirm our key observations with human evaluation.

#### 4.1.2 Human Evaluation

**Quality** We ask a pool of 602 crowdworkers on Amazon Mechanical Turk to evaluate various sampling configurations in the quality aspect. Each worker is presented a set of ten samples along with the prompts (prefixes). They are then asked to rate how likely the sentence would appear in a news article between 0 and 5 (Invalid, Confusing, Unspecific, Average, Expected, and Very Expected respectively).

We focus on the Gigaword dataset for human evaluation since news articles are ubiquitous and do not often require expert knowledge for quality judgement. For each configuration (sampling algorithm and hyperparameter pair) we ask crowdworkers to rate 200 samples in total. To get an accurate rating for each sample, we enlist 25 different crowdworkers to rate each sample. We report mean and standard deviation from 5 independent runs (each with 40 samples) as error bar.

By manual inspection, we find that the time spent in the annotations is a good indicator of the quality

of the rating. Therefore, we estimate the human judgement score for a sample as the average rating of the 20 crowdworkers (out of 25) who took the most time to rate the samples. We provide further details about our setup in Appendix C and D.

**Diversity** It is difficult for human annotators to estimate diversity of text (Hashimoto et al., 2019). Therefore, we use the *n-gram entropy* metric (Zhang et al., 2018; He and Glass, 2019). Given  $S_{\text{gen}}$  which contains a large number of samples, we measure its diversity using the following formulation:

$$\mathcal{H}^{n\text{-gram}}(S_{\text{gen}}) = \sum_{g \in G_n} -r(g) \log r(g), \quad (13)$$

where  $G_n$  is the set of all n-grams that appeared in  $S_{\text{gen}}$ , and  $r(g)$  refers to the ratio (frequency) of n-gram  $g$  w.r.t. all n-grams in the  $S_{\text{gen}}$ . For the estimation of n-gram entropy, we generate 50,000 samples from each sampling configuration.

We will report human quality score either paired with n-gram entropy or with self-BLEU as diversity metric. We find they give similar observations.

## 4.2 Model and Datasets

We separately fine-tune GPT2-small (Radford et al., 2018; Wolf et al., 2019) (110M parameters) on the Gigaword (Graff et al., 2003; Napoles et al., 2012) and the WikiText-103 (Merity et al., 2017) datasets. We use the same tokenization as GPT-2, and add additional padding and end-of-sequence tokens ([EOS]) to the sentences.

To generate a sequence, we feed a length-10 prefix from test data into the fine-tuned GPT-2 model, and use a sampling algorithm to complete the sentence. Since shorter samples are more difficult to judge in quality (Ippolito et al., 2020), we filter all generated sentence completions to be between 40 and 50 subwords, and filter our validation and test set to meet the same requirements. To permit validation and test sets that are large enough to prefix 10,000 sentences for the corpus-BLEU metric, we re-chunk the first 80% of the Gigaword dataset for the training set, 15% for validation, and the last 5% for the test set. Similarly, we re-chunk the first 97% of the WikiText-103 dataset for training, and leave 1.5% for validation and 1.5% for test.

## 5 Empirical Results

First, we compare existing sampling algorithms, and then move on to validate the necessity and

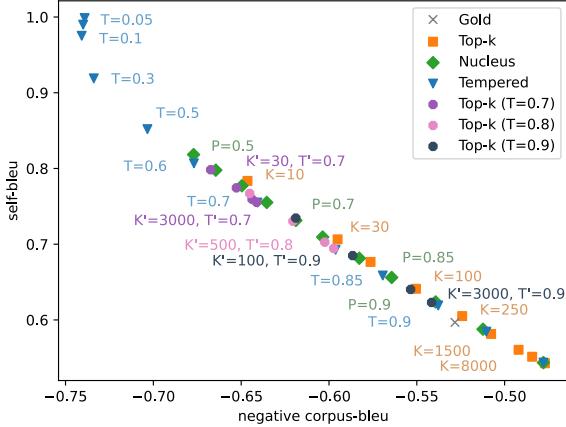


Figure 2: The performance (x-axis: quality, y-axis: diversity, both are the smaller the better) of top- $k$ , nucleus, tempered and tempered top- $k$  sampling are on par on the Gigaword dataset, as shown by automatic evaluation.

sufficiency of the identified properties.

### 5.1 Comparison of Existing Algorithms

We compare top- $k$ , nucleus, and tempered sampling via automatic and human evaluation. We do a fine-grained sweep of hyperparameters for each sampling algorithm on the Gigaword dataset. The results are shown in Figure 1 (human evaluation) and Figure 2 (automatic evaluation). We also show the quality and diversity score for human text in the test data for reference, which is labeled as gold.

Both automatic and human evaluations demonstrate that the performance of top- $k$ , nucleus and tempered sampling are on par with each other, with no significant gap. When the hyperparameters ( $K$ ,  $P$  and  $T$ ) are tuned so that different sampling has the same diversity (measured by self-BLEU or n-gram entropy), their quality (measured by corpus-BLEU or human rating) are close.

Additionally, we compare tempered top- $k$  sampling with the existing algorithm also in Figure 2. We find that adding the tempered transformation only moves top- $k$  sampling along the Q-D trade-off, instead of yielding a better or a worse sampling algorithm. For example, the performance of the  $K = 500, T = 0.8$  configuration for tempered top- $k$  sampling is very close to the  $K = 30$  configuration for the top- $k$  sampling.

Motivated by these observations, we identify three core properties (elaborated in Section 3.1) that are shared among the sampling algorithms: *entropy reduction*, *order preservation* and *slope preservation*. In the following two sections, we

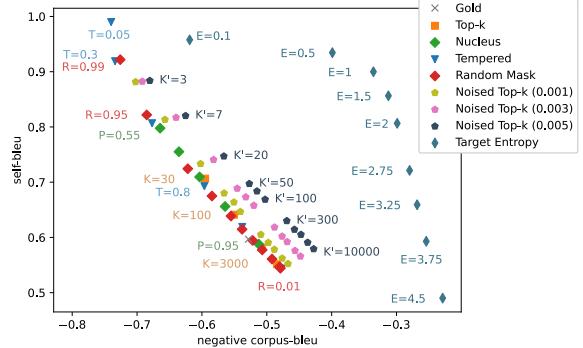


Figure 3: Automatic evaluation of the noised top- $k$ , target entropy, and random mask sampling proposed to validate the necessity of the identified properties. The results show that violation of entropy reduction and slope preservation could lead to drastic performance degradation, while the order preservation property could be further relaxed.

present experiments validating the necessity or sufficiency aspect of the properties.

### 5.2 Property-violating Algorithms

In Figure 3, we compare the generation performance of the property-violating sampling algorithms (designed in Section 3.2), against the existing algorithms using automatic evaluation on the Gigaword dataset. We make the following observations: First, the target entropy sampling, which violates entropy reduction, has significantly worse performance; Second, even with small noise weight  $W$ , the performance of noised top- $k$  sampling degrades from the original top- $k$  sampling, and the gap becomes larger as  $W$  increases; Last, the random mask sampling is on par with the existing sampling algorithms in performance. We further confirm this observation with human evaluation in Figure 5.

These results suggest that the violation of entropy reduction or slope preservation could lead to drastic performance degradation. On the other hand, the competitive performance of random mask sampling suggests that order preservation could be further relaxed.

In the next section, we investigate the sufficiency aspect of the identified properties.

### 5.3 Property-satisfying Algorithms

We now compare the generation performance of the property-satisfying sampling algorithms (designed in Section 3.2) with the existing sampling algorithms. The results from the Gigaword dataset

Sampling	Conditional Samples
<b>Existing Sampling Algorithms</b>	
<i>Top-k</i> ( $K = 30$ )	<i>steven spielberg's dreamworks movie studio</i> said monday it was filing a lawsuit, accusing us studio executives of defrauding hundreds of thousands of dollars in refunds and other damages.
<i>Nucleus</i> ( $P = 0.80$ )	<i>steven spielberg's dreamworks movie studio</i> has failed to attract the kind of business and development investors that jeffrey hutchinson dreamed up in the past.
<i>Tempered</i> ( $T = 0.85$ )	<i>steven spielberg's dreamworks movie studio</i> plans to spend the rest of the year producing the high-speed thriller "the earth's path" and an upcoming sequel, the studio announced on wednesday.
<b>Property-satisfying Sampling Algorithms</b>	
<i>Random Top-k</i> ( $R = 90$ )	<i>steven spielberg's dreamworks movie studio</i> is planning to make a movie about a young man who is a <unk>, a man who has a dream of being the first man to be born with the ability to walk on water.
<i>Max Entropy</i> ( $E = 2.75$ )	<i>steven spielberg's dreamworks movie studio</i> has agreed to pay \$ #.# million to director john nichols (\$ #. million, ###, a record in the studio circulation ), the studio announced sunday..
<b>Property-violating Sampling Algorithms</b>	
<i>Random Mask</i> ( $R = 0.75$ )	<i>steven spielberg's dreamworks movie studio</i> scored a big win with a \$ ##.# million ( euro ##.# million ) direct-to-video ( dvds ) deal to develop the ##### short story "the rose garden".
<i>Noised Top-k</i> ( $K=50$ , $W=5e-3$ )	<i>steven spielberg's dreamworks movie studio</i> is in disarray and has a few directors and a lot of stock involved, leaving it only a matter of time before <i>spielberg's departure from the nobel peace prize</i> .
<i>Target Entropy</i> ( $E = 2.75$ )	<i>steven spielberg's dreamworks movie studio</i> production scored an action <i>boost m boom, nabbing an 'd</i> after the ##th instal specialization with nominations of fritz, ika, ivan english ape and evlyn mcready.

Table 1: Generated sequences with the same prefix *steven spielberg's dreamworks movie studio* by different sampling algorithms. The hyperparameters are chosen such that the algorithms yield roughly the same diversity measured by self-BLEU. The poor-quality spans are highlighted in red.

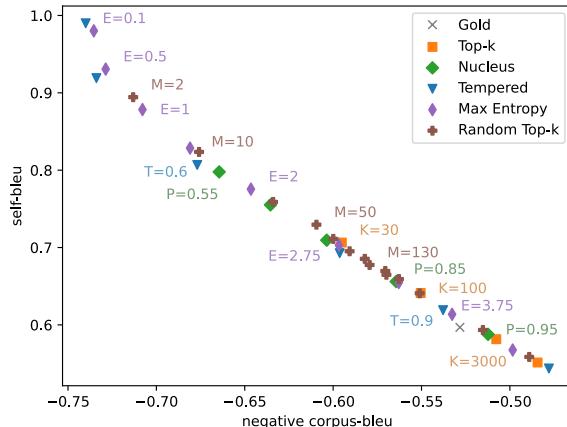


Figure 4: The proposed random top- $k$  and max entropy schedulers, which meet the identified properties, are on par in performance with existing methods in automatic evaluation on the Gigaword dataset.

are shown in Figure 3 (for automatic evaluation) and Figure 5 (for human evaluation). For completeness, we also replicate Figure 5 with self-BLEU as the diversity measure in Appendix F. We also present results from automatic evaluation on the Wikitext-103 dataset in Figure 6, with consistent observations.

The evaluations consistently show that the performance of random top- $k$  and max entropy sampling (and random mask sampling in last section) is on par with top- $k$ , nucleus, and tempered sampling. These results strengthen the importance of the iden-

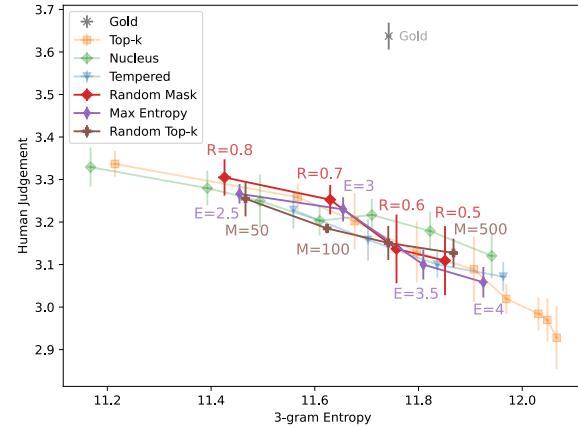


Figure 5: Human evaluation also shows that the proposed sampling algorithms has performance on par with the existing methods on the Gigaword dataset. Appendix F repeats this plot with self-BLEU.

tified properties in that, new sampling algorithms could get competitive generation performance as long as they meet the identified properties.

#### 5.4 Qualitative Analysis

We list samples from the proposed sampling algorithms and compare them with the existing ones in Table 1. We choose the hyperparameter of each sampling algorithm so that each algorithm exhibits a similar level of diversity (as measured by self-BLEU). By manual inspection, we find that the quality of samples from property-satisfying sam-

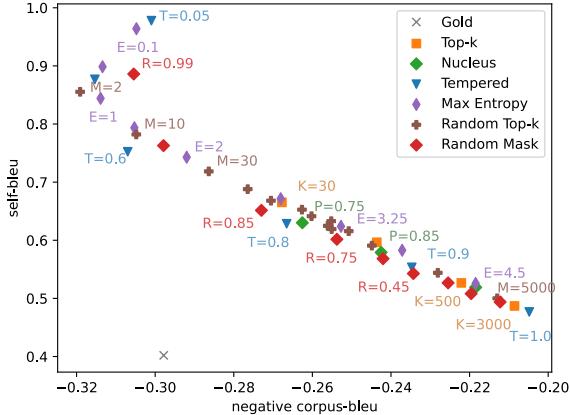


Figure 6: Automatic evaluation on the WikiText-103 dataset: The performance of proposed sampling algorithms are on par with top- $k$ , nucleus, and tempered sampling.

pling algorithms is on par with samples from the existing algorithms. In particular, the samples from random top- $k$ , max entropy, and random masked sampling are all coherent and informative.

In contrast, the samples from noised top- $k$  and target entropy algorithms, tend to be less semantically and syntactically coherent. In particular, the target entropy sampling algorithm, which obtains the lowest quality score measured by corpus-BLEU, lacks basic language structure. In comparison to target entropy, noised top- $k$  is syntactically coherent, but exhibits logical and factual inconsistencies. These observations aligns with the results we get from automatic evaluation.

## 6 Related Works

Despite the popularity of sampling algorithms in natural language generation, a rigorous comparison or scrutiny of existing algorithms is lacking in the literature. Ippolito et al. (2019) provides a comparison between sampling and decoding algorithms. Holtzman et al. (2020) proposes nucleus sampling, and compare it with top- $k$  sampling (Fan et al., 2018). However, only a few hyperparameter configurations are tested. In Hashimoto et al. (2019) and Caccia et al. (2020), temperature sampling is used and the hyperparameter  $T$  is tuned to trade-off between diversity and quality, but it lacks comparisons with other sampling algorithms. Welleck et al. (2020) studies the *consistency* of existing sampling and decoding algorithms, without comparing the generation performance.

In this work we mainly use the quality-diversity trade-off (Caccia et al., 2020) to conduct a compar-

ison of different sampling algorithms. Parallel to our work, Zhang et al. (2020a) also uses the quality-diversity trade-off to compare top- $k$ , nucleus, and tempered sampling. Their observation is similar to ours: The performance of the existing algorithms are close with no significant gap.

More importantly, the underlying reasons for the success of various sampling algorithms remain poorly understood. Zhang et al. (2020a) proposes the *selective* sampling algorithm, which fails to outperform existing approaches. This failed attempt suggests the need for a better understanding of the strengths and weaknesses of existing methods. To the best of our knowledge, our work provides the first systematic characterization of sampling algorithms, where we attribute the success of existing sampling algorithms to a shared set of properties. We show that we can propose novel sampling algorithms based on the identified properties, and reach competitive generation performance as measured by both automatic and human evaluation.

## 7 Limitations and Future Work

Our core contribution is the three properties of sampling algorithms that we conjecture are crucial for competitive generation performance. While we design a set of experiments to validate their necessity and sufficiency, the observations we make are still empirical. We emphasize that **it is completely possible that there exists some crucial property, that is yet to be discovered, and can lead to significantly better generation performance**. Therefore, the exploration of novel sampling algorithms (Zhang et al., 2020a) should still be encouraged.

On the other hand, to provide a comprehensive study, we focus on the open-ended language generation task with the GPT-2 model. As future work, it would be interesting to check whether our observations also hold on other tasks such story generation or dialogue response generation, or with weaker language models in low-resource setting.

## 8 Conclusion

This work studies sampling algorithms for the open-ended language generation task. We show that the existing algorithms, namely top- $k$ , nucleus, and tempered sampling, have similar generation performance as measured by the quality-diversity trade-off evaluation. Motivated by this result, we identify three key properties that we prove are shared by

the existing algorithms. To validate the importance of these identified properties, we design a set of new sampling algorithms, and compare their performance with the existing sampling algorithms. We find that violation of the identified properties may lead to drastic performance degradation. On the other hand, we propose several novel algorithms, namely random top- $k$  and max entropy sampling, that meet the identified properties. We find that their generation performance is on par with the existing algorithms.

## Acknowledgments

The authors sincerely thank Yixin Tao, Jingzhao Zhang and Yonatan Belinkov for useful discussions. This work was partly supported by Samsung Advanced Institute of Technology (Next Generation Deep Learning: from pattern recognition to AI), Samsung Electronics (Improving Deep Learning using Latent Structure). Kyunghyun Cho thanks CIFAR, Naver, eBay, NVIDIA and Google for their support.

This research was sponsored in part by the United States Air Force Research Laboratory and was accomplished under Cooperative Agreement Number FA8750-19-2-1000. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the United States Air Force or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

## References

- David H. Ackley, Geoffrey E. Hinton, and Terrence J. Sejnowski. 1985. *A Learning Algorithm for Boltzmann Machines*, volume 9, pages 147–169.
- Anja Belz and Ehud Reiter. 2006. *Comparing automatic and human evaluation of NLG systems*. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, Trento, Italy. Association for Computational Linguistics.
- Massimo Caccia, Lucas Caccia, William Fedus, Hugo Larochelle, Joelle Pineau, and Laurent Charlin. 2020. *Language gans falling short*. In *Proceedings of the International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.
- Angela Fan, Mike Lewis, and Yann Dauphin. 2018. *Hierarchical neural story generation*. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.
- David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2003. English gigaword. *Linguistic Data Consortium, Philadelphia*, 4(1):34.
- Tatsunori Hashimoto, Hugh Zhang, and Percy Liang. 2019. *Unifying human and statistical evaluation for natural language generation*. In *Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1689–1701, Minneapolis, Minnesota. Association for Computational Linguistics.
- Tianxing He and James R. Glass. 2019. *Negative training for neural dialogue response generation*. *CoRR*, abs/1903.02134.
- Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. *The curious case of neural text degeneration*. In *Proceedings of the International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.
- Daphne Ippolito, Daniel Duckworth, Chris Callison-Burch, and Douglas Eck. 2020. *Automatic detection of generated text is easiest when humans are fooled*. In *Proceedings of the Association for Computational Linguistics*, pages 1808–1822, Online. Association for Computational Linguistics.
- Daphne Ippolito, Reno Kriz, Joao Sedoc, Maria Kustikova, and Chris Callison-Burch. 2019. *Comparison of diverse decoding methods from conditional language models*. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3752–3762, Florence, Italy. Association for Computational Linguistics.
- Jiwei Li, Will Monroe, Tianlin Shi, Sébastien Jean, Alan Ritter, and Dan Jurafsky. 2017. *Adversarial learning for neural dialogue generation*. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 2157–2169, Copenhagen, Denmark. Association for Computational Linguistics.
- Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In *Proceedings of European Conference on Computer Vision*.
- Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. *Pointer sentinel mixture models*. In *Proceedings of the International Conference on Learning Representations*. OpenReview.net.
- Tomas Mikolov, Martin Karafiat, Lukás Burget, Jan Černocký, and Sanjeev Khudanpur. 2010. *Recurrent neural network based language model*. In *Proceedings of the International Speech Communication Association*, pages 1045–1048.

- Courtney Napoles, Matthew Gormley, and Benjamin Van Durme. 2012. *Annotated Gigaword*. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX)*, pages 95–100, Montréal, Canada. Association for Computational Linguistics.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. *Bleu: a method for automatic evaluation of machine translation*. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.
- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2018. *Language models are unsupervised multitask learners*.
- Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. *Neural machine translation of rare words with subword units*. In *Proceedings of the Association for Computational Linguistics*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.
- Sean Welleck, Ilia Kulikov, Jaedeok Kim, Richard Yuanzhe Pang, and Kyunghyun Cho. 2020. *Consistency of a recurrent language model with respect to incomplete decoding*. *CoRR*, abs/2002.02492.
- Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierrette Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. *Huggingface’s transformers: State-of-the-art natural language processing*. *CoRR*, abs/1910.03771.
- Lijun Wu, Yingce Xia, Fei Tian, Li Zhao, Tao Qin, Jianhuang Lai, and Tie-Yan Liu. 2018. *Adversarial neural machine translation*. In *Proceedings of The Conference on Machine Learning*, volume 95 of *Proceedings of Machine Learning Research*, pages 534–549. PMLR.
- Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. 2018. *Recent trends in deep learning based natural language processing [review article]*. *IEEE Comput. Intell. Mag.*, 13(3):55–75.
- Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2016. *Seqgan: Sequence generative adversarial nets with policy gradient*. *CoRR*, abs/1609.05473.
- Hugh Zhang, Daniel Duckworth, Daphne Ippolito, and Arvind Neelakantan. 2020a. *Trading off diversity and quality in natural language generation*.
- Yizhe Zhang, Michel Galley, Jianfeng Gao, Zhe Gan, Xiuju Li, Chris Brockett, and Bill Dolan. 2018. *Generating informative and diverse conversational responses via adversarial information maximization*.
- In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Proceedings of Neural Information Processing Systems 31*, pages 1810–1820. Curran Associates, Inc.
- Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020b. *DIALOGPT : Large-scale generative pre-training for conversational response generation*. In *Proceedings of the Association for Computational Linguistics: System Demonstrations*, pages 270–278, Online. Association for Computational Linguistics.
- Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. *Txygen: A benchmarking platform for text generation models*. In *Proceedings of the Conference on Research & Development in Information Retrieval*, pages 1097–1100. ACM.

## A Auxiliary Plots

We show the importance of preserving the token with the largest probability ( $p_1$ ) in the proposed random mask sampling. For comparison, we relax the constraint and define the *random mask-all* sampling:

**Definition A.1. (Random Mask-all)** The only difference between random mask-all sampling and random mask sampling is that we allow the  $p_1$  token to be masked. We formulate it below:

$$\hat{p}_i = \frac{p'_i}{\sum_{j=1}^{|V|} p'_j}, \quad (14)$$

where  $p'_i = p_i \cdot \mathbb{1}\{u_i > R\}$  and  $u_i \sim U(0, 1)$ .

In Figure 7, we show that if  $p_1$  is allowed to be masked, the generation performance will be seriously degraded.

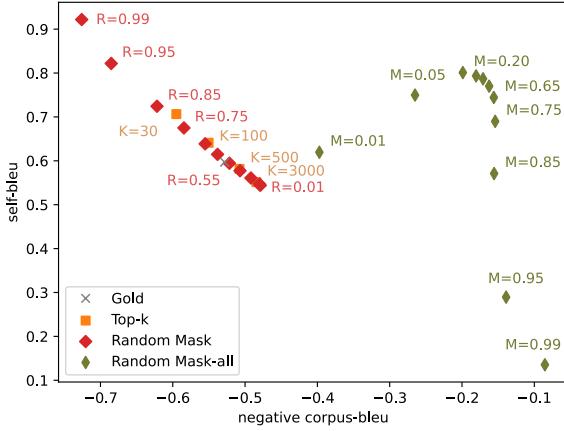


Figure 7: The random mask-all sampling, where  $p_1$  is allowed to be masked, is shown to have worse performance than the random mask sampling. The dataset is Giagword.

## B Proof for Proposition 1

In this section we prove Proposition 1.

Firstly, it is straightforward to prove that Property 2 (order preservation) holds for the top- $k$ , nucleus and tempered sampling and we omit the proof here.

For Property 3 (slope preservation), it holds trivially for nucleus and top- $k$  sampling. We prove it for tempered sampling in the following lemma:

**Lemma B.1.** Property 3 holds for tempered sampling (Definition 2.3).

*Proof.* Remember that the tempered sampling with hyperparameter  $T$  defines the follow transformation:  $\hat{p}_i = \frac{p'_i}{\sum_j p'_j}$ , where  $p'_i = \exp(\log(p_i)/T)$ .

We set  $Z = \sum_j p'_j$ , then  $\forall \hat{p}_i > \hat{p}_j > \hat{p}_k > 0$  we have

$$\begin{aligned} & \frac{\log \hat{p}_i - \log \hat{p}_j}{\log \hat{p}_j - \log \hat{p}_k} \\ &= \frac{\log p'_i - \log Z - \log p'_j + \log Z}{\log p'_j - \log Z - \log p'_k + \log Z} \\ &= \frac{\log p'_i - \log p'_j}{\log p'_j - \log p'_k} \quad (\text{log } Z \text{ is cancelled}) \\ &= \frac{\log(p_i)/T - \log(p_j)/T}{\log(p_j)/T - \log(p_k)/T} \\ &= \frac{\log(p_i) - \log(p_j)}{\log(p_j) - \log(p_k)} \end{aligned} \quad (15)$$

□

Only Property 1 (entropy reduction) is left. We now prove it holds for top- $k$  / nucleus sampling:

**Lemma B.2.** Property 1 holds for transformations defined by top- $k$  or nucleus sampling (Definition 2.1 and 2.2).

*Proof.* We first consider the change of entropy when the token with the smallest probability ( $p_{|V|}$ ) is removed from the original distribution ( $\hat{p}_i = \frac{p_i}{\sum_{j=1}^{|V|-1} p_i}$ ,  $1 \leq i < |V|$ ):

$$\begin{aligned} -\mathcal{H}(\mathbf{p}) &= \sum_{i=1}^V p_i \log p_i \\ &= \sum_{i=1}^{V-1} p_i \log p_i + p_{|V|} \log p_{|V|} \\ &= (1 - p_{|V|}) \sum_{i=1}^{V-1} \frac{p_i}{1 - p_{|V|}} \log p_i + p_{|V|} \log p_{|V|} \\ &= \sum_{i=1}^{V-1} \frac{p_i}{1 - p_{|V|}} \log \frac{p_i}{1 - p_{|V|}} + \underbrace{\log(1 - p_{|V|})}_{<0} \\ &\quad + p_{|V|} \left( \log p_{|V|} - \sum_{i=1}^{V-1} \frac{p_i}{1 - p_{|V|}} \log p_i \right) \\ &< \sum_{i=1}^{V-1} \hat{p}_i \log \hat{p}_i + p_{|V|} \left( \log p_{|V|} - \sum_{i=1}^{V-1} \frac{p_i}{1 - p_{|V|}} \log \underbrace{\frac{p_i}{1 - p_{|V|}}}_{>p_{|V|}} \right) \\ &< \sum_{i=1}^{V-1} \hat{p}_i \log \hat{p}_i + p_{|V|} \left( \log p_{|V|} - \underbrace{\sum_{i=1}^{V-1} \frac{p_i}{1 - p_{|V|}} \log p_{|V|}}_{=\log p_{|V|}} \right) \\ &= \sum_{i=1}^{V-1} \hat{p}_i \log \hat{p}_i = -\mathcal{H}(\hat{\mathbf{p}}) \end{aligned} \quad (16)$$

Therefore, we get  $\mathcal{H}(\hat{\mathbf{p}}) < \mathcal{H}(\mathbf{p})$ .

By induction (iteratively removing the last token), it is now easy to see that the top- $k$  or nucleus

transformation strictly decrease the entropy of the sampling distribution.  $\square$

Finally, we prove Property 1 (entropy reduction) holds for tempered sampling:

**Lemma B.3.** Property 1 holds for the transformation defined by tempered sampling (Definition 2.3).

*Proof.* For convenience, we first rewrite the Temperature transformation:

$$\hat{p}_i = p_i^\alpha = \frac{\exp(-\alpha e_i)}{\sum_j \exp(-\alpha e_j)} \quad (17)$$

where  $e_i = -\log(p_i)$  and  $\alpha = \frac{1}{T}$ . The entropy can be written as:

$$\begin{aligned} \mathcal{H}(p^\alpha) &= -\sum_i \frac{\exp(-\alpha e_i)}{\sum_j \exp(-\alpha e_j)} \log \frac{\exp(-\alpha e_i)}{\sum_j \exp(-\alpha e_j)} \\ &= \log \sum_j \exp(-\alpha e_j) + \alpha \sum_i e_i \frac{\exp(-\alpha e_i)}{\sum_j \exp(-\alpha e_j)} \end{aligned} \quad (18)$$

Next, we take derivative w.r.t  $\alpha$ :

$$\begin{aligned} \frac{\partial \mathcal{H}}{\partial \alpha} &= -\underbrace{\sum_i e_i \frac{\exp(-\alpha e_i)}{\sum_j \exp(-\alpha e_j)}}_{=0} + \sum_i e_i \frac{\exp(-\alpha e_i)}{\sum_j \exp(-\alpha e_j)} \\ &\quad + \alpha \frac{\partial}{\partial \alpha} \sum_i e_i \frac{\exp(-\alpha e_i)}{\sum_j \exp(-\alpha e_j)} \\ &= \alpha \sum_i e_i \left[ \frac{\partial}{\partial \alpha} \log \frac{\exp(-\alpha e_i)}{\sum_j \exp(-\alpha e_j)} \right] \left[ \frac{\exp(-\alpha e_i)}{\sum_j \exp(-\alpha e_j)} \right] \\ &\quad \text{log-derivative trick} \\ &= \alpha \sum_i e_i \left[ -e_i + \sum_{j'} e_{j'} \frac{\exp(-\alpha e_i)}{\sum_j \exp(-\alpha e_j)} \right] \\ &\quad \left[ \frac{\exp(-\alpha e_i)}{\sum_j \exp(-\alpha e_j)} \right] \\ &= -\alpha \mathbb{E}_{p^\alpha} [e_i^2] + \mathbb{E}_{p^\alpha} [e_i] \\ &= -\underbrace{\alpha}_{>0} \underbrace{\left( \mathbb{E}_{p^\alpha} [e_i^2] - \mathbb{E}_{p^\alpha} [e_i]^2 \right)}_{=\text{Var}_{p^\alpha} [e_i] \geq 0} \\ &< 0 \end{aligned} \quad (19)$$

We can now easily get  $\frac{\partial \mathcal{H}}{\partial T} = \frac{\partial \mathcal{H}}{\partial \alpha} \frac{\partial \alpha}{\partial T} > 0$ . Therefore, when we apply a tempered transformation with  $T < 1$ , the entropy will strictly decrease comparing to the original distribution (where  $T = 1$ ).  $\square$

## C Mechanical Turk Setup

Our crowdworkers were required to have a HIT acceptance rate higher than 95%, and be located

in the United States. In total, 602 crowdworkers completed our tasks. In order to ensure that we had quality data, we filtered the crowdworker annotations for workers that spent at least 45 seconds on the aggregate task (or 4.5 seconds rating each sentence). 51 crowdworkers were filtered out through this process. Screenshots of our instructions and task are available in Figure(s) 8 and 9 respectively.

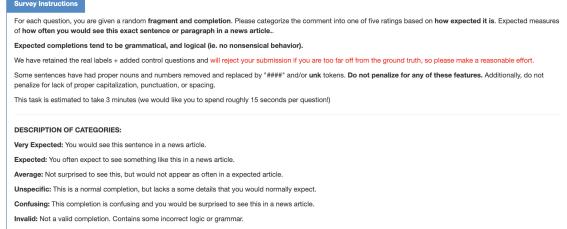


Figure 8: Our instructions for crowdworker task.

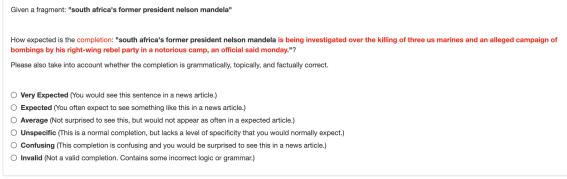


Figure 9: An example of the task given to crowdworkers.

## D Convergence of Human Evaluation

When we conduct human evaluation, we provide crowdworkers with 200 generated samples for some configuration, and ask 25 different crowdworkers to evaluate the same sample. However, a reasonable question is whether our human evaluations are converging to some underlying true rating, or whether we need more samples or replicas.

Figure 10 and 11 show that the average scores have roughly converged around 150 samples per configuration, or around 15 replicas per sample. The two figures demonstrate this for nucleus sampling, and this holds true for human evaluations of all sampling algorithms.

## E Additional Model-Generated Samples

Table 2 shows some additional samples from each of the sampling algorithms described in the paper. Similarly, we have chosen hyperparameters for each sampling method that yields a similar diversity (measured by self-BLEU) to the top- $k$  configuration where  $K = 15$ . We observe that all sampling

Sampling	Conditional Samples
<b>Existing Sampling Algorithms</b>	
Top-K ( $K = 15$ )	as the rest of his denver broncos teammates prepared for the game against denver, jay kasey could not help but think of his teammates and friends who worked hard in preparation for that night's game.
Nucleus ( $P = 0.65$ )	as the rest of his denver broncos teammates slumped and buried themselves in their work, broncos quarterback leon johnson moved to the locker room monday and called his parents.
Temperature ( $T = 0.7$ )	as the rest of his denver broncos teammates gathered in an auditorium to watch more stretching drills, ben holtz gave an emotional speech : we're running out of time to win a championship ring.
<b>Property-satisfying Sampling Algorithms</b>	
Random Top-K ( $R = 30$ )	as the rest of his denver broncos teammates battled through their own stretch of the nfl playoffs, the quarterback began throwing the ball in the fourth quarter.
Max Entropy ( $E = 2.75$ )	steven spielberg's dreamworks movie studio has agreed to pay \$ #.# million to director john nichols (\$ #. million, ###, a record in the studio circulation ), the studio announced sunday..
<b>Property-violating Sampling Algorithms</b>	
Random Mask ( $R = 0.75$ )	as the rest of his denver broncos teammates connect with a player that the team didn't expect to become a starter, quarterback james crosby speaks out about colin peterson's passion for the game.
Noised Top-K ( $K=20, W=5e-3$ )	as the rest of his denver broncos teammates start making room for nerdy bundles or twiggy pitchers, coach william perez might have to cut a big, bold note cut ready to console wife join them in iraq.
Target Entropy ( $E = 2.5$ )	as the rest of his denver broncos teammates scratched out their locker rooms, cleanDeath Yo Communities wander edge extingustretched cords429 Mohnegie wildfires.

Table 2: The samples conditioned on *as the rest of his denver broncos teammates*, and the hyperparameters for a given sampling algorithm. The poor quality spans are highlighted in red.

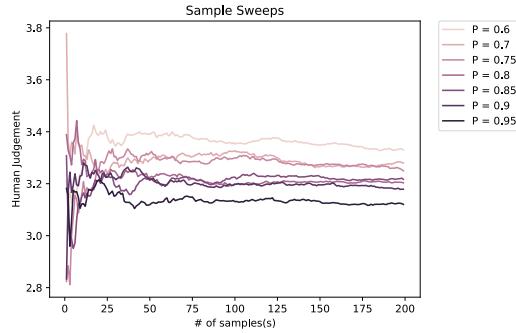


Figure 10: We see that we obtain a reasonable estimate of sample quality around 150 samples per configuration.

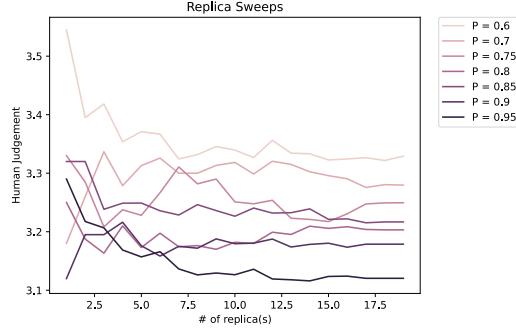


Figure 11: We see that we obtain a reasonable estimate of sample quality with around 15 ratings per sample.

algorithms except for noised top- $k$  and target entropy, yield similar quality samples. For noised top- $k$  and target entropy, we see that these samples tend to degenerate towards the end of the sentence,

indicating violation of the identified properties may possibly lead towards degraded performance.

## F Human Evaluation with Self-BLEU as Diversity Metric

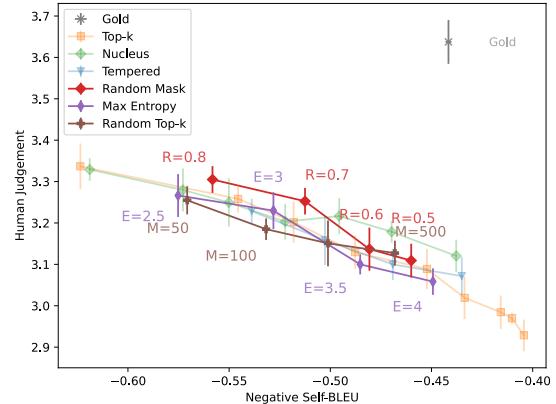


Figure 12: Using self-BLEU as a diversity metric provides similar conclusions as to using n-gram entropy.

Figures 1 and 5 measures diversity in terms of 3-gram entropy, while the rest of our work measures diversity in terms of self-BLEU. For completeness, we provide Figure 12 where self-BLEU is used for diversity metric. This figure demonstrates that similar trends can be observed using either 3-gram entropy or self-BLEU.

# Chinese Content Scoring: Open-Access Data Sets and Features on Different Segmentation Levels

**Yuning Ding Andrea Horbach Haoshi Wang Xuefeng Song Torsten Zesch**  
Language Technology Lab, University Duisburg-Essen  
(yuning.ding|andrea.horbach|torsten.zesch)@uni-due.de

## Abstract

In this paper, we analyse the challenges of Chinese content scoring in comparison to English. As a review of prior work for Chinese content scoring shows a lack of open-access data in the field, we present two short-answer data sets for Chinese. The Chinese Educational Short Answers data set (CESA) contains 1800 student answers for five science-related questions. As a second data set, we collected ASAP-ZH with 942 answers by re-using three existing prompts from the ASAP data set.

We adapt a state-of-the-art content scoring system for Chinese and evaluate it in several settings on these data sets. Results show that features on lower segmentation levels such as character n-grams tend to have better performance than features on token level.

## 1 Introduction

Short answer questions are a type of educational assessment that requires respondents to give natural language answers in response to a question or some reading material (Rademakers et al., 2005). The applications used to automatically score such questions are usually thought of as content scoring systems, because content (and not linguistic form) is taken into consideration for automatic scoring (Zhai et al., 2012). While there is a large research body for English content scoring, there is less research for Chinese.<sup>1</sup> The largest obstacle for more research on Chinese is the lack of publicly available data sets of Chinese short answer questions.

<sup>1</sup>In this work, we use the term ‘Chinese’ as abbreviation for Mandarin Chinese, which includes simplified and traditional written Chinese. Cantonese, Wu, Min Nan and other dialects are not included.

Working with Chinese poses substantially different challenges than work on English data. Unlike English, which uses spaces as natural separators between words, segmentation of Chinese texts into tokens is challenging (Chen and Liu, 1992). Furthermore, there are more options on which level to segment Chinese text. Apart from tokenization and segmentation into characters, which are two options also available and often used for English, segmentation into components, radicals and even individual strokes are additionally possible for Chinese. Table 1 gives an example for the segmentation options in both languages. Orthographic variance can be challenging in both languages, but behaves very differently. Non-word errors, which is the main source of orthographic problems in English (Mitton, 1987), can by definition not happen in Chinese, due to the input modalities.

Language	Level	Unigrams
English	word	panda
	characters	p, a, n, d, a
Chinese	word	熊猫
	characters	熊, 猫
	components	能, ..., 犬, 苗
	radicals	灑, 犭
	strokes	𠵼、𠂇--...

Table 1: Comparison of segmentation possibilities in English and Chinese

In the remainder of this paper, we will discuss these challenges in more detail (Section 2). We review prior work on Chinese content scoring (Section 3) and present two new freely-available data sets of short answers in Chinese (Section 4). In Section 5, we adapt a

machine learning pipeline for automatic scoring with state-of-art NLP tools for Chinese. We investigate the extraction of n-gram features on all possible segmentation levels. In addition, we use features based on the Pinyin transcription of Chinese texts and experiment with the removal of auxiliary words as an equivalent to lemmatization in English. We evaluate these features on our new data sets as well as, for comparison, an English data set translated into Chinese.

## 2 Challenges in Chinese Content Scoring

In this section, we highlight the main challenges when processing Chinese learner data in comparison to English data sets. We first focus on segmentation, as tokenization is more difficult in Chinese than in English and there are more linguistic levels on which to segment a Chinese text compared to English. Next, we discuss variance in learner answers, which is a challenge for content scoring in any language but manifests itself in Chinese differently than in English.

### 2.1 Segmentation

English has an alphabetic writing system with some degree of grapheme-to-phoneme correspondence. The Chinese language, in contrast, uses a logosyllabic writing system, where characters represent lexical morphemes. Chinese words can be formed by one or more characters (Chen, 1992). Unlike English, where words are separated by white-spaces, the fact that Chinese writing does not mark word boundaries makes word segmentation a much harder task in Chinese NLP (e.g., Chen and Liu (1992); Huang et al. (1996)). According to a recent literature review on Chinese word segmentation (Zhao et al., 2019), the best-performing segmentation tool has an average F1-value of only around 97%. A major challenge is the handling of out-of-vocabulary words.

In English content scoring, word level features such as word n-grams or word embeddings have proven to be effective (e.g., Sakaguchi et al. (2015); Riordan et al. (2017)). Additionally, character features are frequently used to capture orthographic as well as morphological variance (e.g., Heilman and Mad-

nani (2013); Zesch et al. (2015)).

In the light of the tokenization challenges mentioned above, it is surprising that although most prior work on Chinese also applies word-level features (see Section 3), the performance of their tokenizers are barely discussed and character-level features are neglected altogether.

Apart from words and characters, there are more possibilities of segmentation in Chinese as discussed above. Consider, for example, a <sup>panda bear</sup> Chinese bi-morphemic word such as 熊猫 . It can additionally be segmented on the stroke, component and radical level as shown in Table 1.

It has been argued that the morphological information of characters in Chinese consists of the sequential information hidden in stroke order and the spatial information hidden in character components (Tao et al., 2019). Each Chinese character can directly be mapped into a series of strokes (with a particular order). On the component level, it has been estimated that about 80% of modern Chinese characters are phonetic-logographic compounds, each of which consists of two components: One carries the sound of the character (the stem) and the other the meaning of the character (the radical) (Li, 1977). We argue that, together with strokes, both kinds of components may be used as features in content scoring. Note that in some cases, a character has only one component, which in the extreme case consists of one stroke only, so that for the character <sup>one</sup> , all four segmentation levels yield the same result, somewhat comparable to an English one-character word, such as “I”.

### 2.2 Linguistic Variance

Variance in learner answers has a major influence on content scoring performance (Horbach and Zesch, 2019), i.e., the more variance between the answers to a specific prompt, the harder it is to score automatically. If we ignore cases of conceptually different answers, variance means different realizations with approximately the same semantic meaning. As shown in Table 2, if we have a question about the eating habits of pandas, Chinese short answers can contain similar variance as in English, which is realized as both orthographic

variance caused by spelling errors as well as variance of linguistic expression. Note that these types of variance should not influence the score of an answer as it depends only from the content of the answer. Both types of variance are further discussed in the following.

**Spelling errors** in English can be classified into non-word and real-word spelling errors. In our example, “bamboo” is a non-word, while “beer” is a real word spelling error. Both error types occur frequently in English short answer data sets, with non-word errors being more frequent (Mitton, 1987, 1996). A content scoring system must therefore be able to generalize by taking variance in spelling into account (Leacock and Chodorow, 2003). To do so, many systems for English data use character-level features (Heilman and Madnani, 2013; Horbach et al., 2017), such that “bamboo” and “bambu”, while being different tokens, share, for example, the character 3-grams ‘bam’ and ‘amb’ .

For Chinese, the situation is entirely different. Non-word spelling errors are rare and even impossible for digitized data because of the input modalities typically used for Chinese text. When entering a Chinese text on the computer, a writer would normally type the phonetic transcription Pinyin, which is the Romanization of Chinese characters based on their pronunciation. After typing a Pinyin, the writer is shown all corresponding characters from which they choose the right one. As this selection list contains only valid Chinese characters, non-word errors cannot occur by definition. Even if the original data set was collected in hand-written format, the transcription process forces transcribers to correct any non-word error that might occur in the data. For example, if the learner accidentally wrote 熊猫 as 熊綈, the transcriber has no choice but to correct such an error, since the non-word character simply does not exist in the Chinese character set.

There are two steps in the writing / transcription process where errors can still occur: typing letters to spell a Pinyin and choosing a character out of a list for this Pinyin. Previous experiments showed that people usually do not check Pinyin for errors, but wait until the Chinese characters start to show up (Chen

and Lee, 2000). This behaviour generates two types of real-word spelling errors. In our example, spelling errors like confusing 穷 (qiōng) with 熊 (xiōng) are normally caused by wrong letters typed in the first step. The other error type, i.e., choosing a wrong word from the homophones, leads to spelling errors like 珠子(zhū zi) instead of 竹子(zhú zi). Researchers found that nearly 95% of errors are due to the misuse of homophones (Yang et al., 2012), i.e., are errors of the second type. In order to reduce the influence of these errors in content scoring, introducing features presented as Pinyin might be beneficial.

**Variance of linguistic expression** is obviously found in both English and Chinese short answers. As shown in Table 2, nearly the same content can be expressed using different lexical and syntactic choices. Human annotators can usually abstract away from these differences and treat all answers the same. However, linguistic variance is a challenge for automatic scoring systems.

In English content scoring, lemmatization is often considered a useful method to reduce part of the variance (Koleva et al., 2014). In this process, words are reduced to their base forms, such as substituting “ate” with “eat” and deleting the “s” after “bamboo”. In Chinese, similar grammatical morphemes such as “了” and “们”, termed auxiliary words (Zan and Zhu, 2009), which indicate the past tense and plural, can also be deleted in a pre-processing step to achieve a similar effect.

Another type of variance is caused by synonyms. For such cases of lexical variance, external knowledge is often needed to decide that two different words are interchangeable. However, as we can see in Table 2, some synonyms, such as “panda bears” vs. “pandas” and 竹子 vs. 竹 share some character(s). Such similarities can be covered by character features, but not token n-grams.

In summary, there is the challenge of the segmentation of Chinese texts into tokens. Features extracted on other segmentation levels might be more robust and therefore helpful for automatic scoring. At the same time, NLP techniques which are useful to reduce variance

	English	Chinese
Reference Answer	Panda bears eat bamboo.	panda bear eat bamboo 熊猫 吃 竹子。
Orthographic Variance	Panda <u>beers</u> eat <u>bambu</u> .	poor cat eat pearl 穷 猫 吃 珠子。
Expression Variance	Panda bears <u>ate</u> <u>bamboos</u> .  Pandas eat bamboo.	panda bear eat <grammatical morpheme for past tense> 熊猫 吃 过 bamboo <grammatical morpheme for plural> 竹子 们 panda bear eat bamboo 熊猫 吃 竹。

Table 2: Example answers showing variance in English and Chinese for the question: *What do panda bears eat?*

in English, especially lemmatization, have not yet been transferred to Chinese. Thus, we will explore in our experiments both n-gram features on different levels and the removal of auxiliary words.

### 3 Prior Work on Chinese Content Scoring

As shown in Table 3, all prior work on Chinese content scoring uses lexical features on the word level, such as word n-grams and sentence length in tokens. They are not only used in shallow learning methods like support vector machines (SVM) or support vector regression (SVR) (Wang et al., 2008; Wu and Shih, 2018), but also applied to deep learning methods like long-short term memory recurrent neural networks (LSTM) (Yang et al., 2017; Huang et al., 2018) or deep autoencoders (Yang et al., 2018).

Also for neural models using word embeddings, word-level tokenization is necessary. Wu and Yeh (2019) train 300-dimensional word2vec word embeddings on sentences from their data set along with Chinese Wikipedia articles and classify student answers with a convolution neural network (CNN). Li et al. (2019) use a Bidirectional Long Short-Term Memory (Bi-LSTM) network for semantic feature extraction from pre-trained 300-dimensional word embeddings (Li et al., 2018) and score student answers based on their similarity to the reference answer using a mutual attention mechanism.

For segmentation, most prior work uses the jieba tokenizer<sup>2</sup> for pre-processing. However,

the performance of the tokenization is rarely discussed. We also notice that no related work uses segmentation on character or component level. Yang et al. (2018) perform stop word removal, but they do not mention if it included some kind of removal of grammatical markers.

### 4 Chinese Scoring Data Sets

In this section, we review existing Chinese content scoring data sets. They are not publicly available, which is a major obstacle to reproducibility in the field. We thus produce two new Chinese data sets (see detailed description in Section 4.2), which are available online<sup>3</sup> to foster future research .

#### 4.1 Existing Data Sets

Horbach and Zesch (2019) give an overview of publicly available data sets for content scoring, five of which are for English, and compare them based on properties such as prompt type, learner population and data set size.

Unfortunately, we did not find any freely available Chinese content scoring data sets. Since we could not access the data sets used in related work, we can only compare them based on their brief descriptions, according to the aspects of comparison mentioned above. Results are shown in Table 4.

The Debris Flow Hazard (DFH) data set is used in the earliest work. It contains more than 1000 answers for 2 prompts in a creative problem-solving task. The learner population are high-school students from Taiwan, who speak native Chinese (Wang et al., 2008).

<sup>2</sup><https://github.com/fxsjy/jieba>

<sup>3</sup><https://github.com/ltdude/ChineseShortAnswerDatasets>

Reference	Data Set	Preprocessing	Features	Classifier	Evaluation
Wang et al. (2008)	DFH task	tokenization, POS tagging	word uni-/bigrams, POS bigrams	SVM	r=.92
Wu and Shih (2018)	SCB-ZH <sup>MT</sup> CS-EN <sup>MT</sup>	tokenization (jieba)	sentence length, word unigrams, BLEU score	SVR, SVM	acc=.60 RMSE=1.17
Yang et al. (2017)	CRCC	tokenization (jieba)	word unigrams	LSTM	acc=.76, Cohen's $\kappa$ =.61
Yang et al. (2018)	CRCC	punctuation and stop word removal, tokenization (jieba)	word unigrams	Auto- encoder	acc=.74, qwk=.64
Huang et al. (2018)	CRCC	tokenization (jieba)	word vector trained on CBOW	LSTM	acc=.74, qwk=.62
Wu and Yeh (2019)	ML_SQA SCB-ZH <sup>MT</sup>	tokenization (jieba)	300D pre-trained word embedding	CNN	acc=.91, recall=.82
Li et al. (2019)	Law Questions	tokenization	300D pre-trained word embedding	Bi-LSTM	acc=.88

Table 3: Overview of related work in Chinese content scoring.

The Chinese Reading Comprehension Corpus (CRCC) (Yang et al., 2018), contains five reading comprehension questions. Each question has on average 2500 answers from students in grade 8.

Instead of collecting and annotating a data set from scratch, Wu and Shih (2018) translated the English SciEntBank (Dzikovska et al., 2013) and the computer science (CS) (Mohler and Mihalcea, 2009) data sets to Chinese. The data set was first translated using machine translation. In order to solve word usage and grammar problems, 12% of the sentences were manually corrected. In their most recent work, the authors also collected a data set with 12 short answer questions and overall 600 answers related to machine learning (ML\_SQA) to compare with the CS-ZH<sup>MT</sup> data set (Wu and Yeh, 2019).

In the most recent work (Li et al., 2019), a large data set containing 85.000 student and reference answers was collected from a national specialty examination related to law.

## 4.2 Collection of Open-access Data Sets

As part of the contribution in this paper, we collected two new data sets for Chinese content scoring: Chinese Short Answer (CESA) and ASAP-ZH. In addition, we provide a machine-translated version of the the original

ASAP-SAS English data, ASAP-ZH<sup>MT</sup>. Table 4 shows key properties, while Table 5 gives example answers of each data set.

**Chinese Educational Short Answers (CESA)** contains five questions from the physics and computer science domain (see Table 6). Answers are collected from 360 students in the computer science department of Zhengzhou University. Each participant was required to answer each question with a maximum of 20 characters, resulting in an average answer length of 13.5 characters. Two annotators speaking native Chinese with computer science background scored the answers into three classes, 0, 1 and 2 points, with an average inter-annotator agreement of 0.9 quadratically weighted kappa (QWK).

**ASAP-ZH** This data set is based on the ASAP short-answer scoring data set released by the Hewlett Foundation.<sup>4</sup> ASAP contains ten short answer prompts covering different subjects and about 2000 student answers per prompt. Prompt 1, 2 and 10 are science-related tasks, which do not have a strong cultural background, and are therefore considered as appropriate to be transferred to other languages.

Therefore, we collected answers in Chinese

<sup>4</sup><http://www.kaggle.com/c/asap-sas>

Data Set	Type	#Answers	#Prompts	Labels	Level
DFH	creative problem solving	2,698	2	[0,1,...,28]	high school
CRCC	reading comprehension	12,528	5	[0,1,2,3,(4),(5)]	middle school
SciEntsBank-ZH <sup>MT</sup>	science	9,804	197	binary&diagnostic	high school
CS-ZH <sup>MT</sup>	computer science	630	21	[0, 0.5,..., 5]	university
ML_SQA	computer science	608	12	binary	university
Law Questions	law	85,000	2	[0,1,5,3]/[0,1,1.5]	-
CESA	physics, computer science	1,800	5	[0,1,2]	university
ASAP-ZH	science	942	3	[0,1,2,(3)]	high school
ASAP-ZH <sup>MT</sup>	science	6,119	3	[0,1,2,(3)]	high school

Table 4: Chinese content scoring data sets: data sets from previous work (upper part) and our new data sets (lower part)

for these three prompts after manually translating the prompt material. The data collection provider BasicFinder<sup>5</sup> helped us to collect 942 answers altogether, 314 answers for each prompt. They are collected from students in high school from grades 9-12, which is comparable with the set of English answers in the ASAP-SAS data set. The answers are transcribed into digital form manually after being collected in handwriting. After reaching an acceptable agreement on a set of answers from the original ASAP-SAS, two annotators speaking native Chinese scored the ASAP-ZH data on a scale from 0 to 3 points (prompt 1 and 2) or 0 to 2 points (prompt 10) with an average QWK of 0.7. Key statistics for the data set can be found in Table 7.

**ASAP-ZH<sup>MT</sup>** For comparison, we also translated the English answers in prompts 1,2 and 10 in the original ASAP-SAS data set to Chinese using the Google Translate API.<sup>6</sup> The examples in Table 5 show that some translation errors can be found, especially when errors exist already in the original text. Words containing spelling errors like “wat” instead of “what” are simply not translated at all. The overall translation quality is also not perfect, for example, the word “coolest” is wrongly translated into <sup>most fashioned</sup> 最 酷的 instead of the correct <sup>most coldest</sup> 最 冷的.

<sup>5</sup><https://www.basicfinder.com/en>

<sup>6</sup><https://cloud.google.com/translate>

As shown in Tables 7 and 8, the average length of the translated answers is larger than the length of the original Chinese answers to the same prompt in our re-collected data set. One explanation could be that paid crowd workers are less motivated than actual students and therefore write shorter answers.

## 5 Experimental Setup

In this section, we adapt a state-of-the-art content scoring system to Chinese. We evaluate it in six settings with different feature sets on the data sets described above in order to investigate different options for segmentation of Chinese text. Table 9 gives an example for the different segmentation options, which will also be detailed in Section 5.2. Additionally, we add a pre-processing step to remove all auxiliary words in the data in order to simulate the effect of lemmatization in English content scoring.

### 5.1 General Experimental Setup

For all our experiments, we use the ESCRITO (Zesch and Horbach, 2018) toolkit and extended it with readers and tokenization for Chinese text. ESCRITO is a publicly available general-purpose scoring framework based on DKPro TC (Daxenberger et al., 2014), which uses an SVM classifier (Cortes and Vapnik, 1995) using the SMO algorithm as provided by WEKA (Witten et al., 1999). For all kinds of features, we use the top 10000 most frequent

Data Set	ID	Score	Example
CESA	5	2	The machine summarizes a large amount of data and finds the pattern from it 机器总结大量数据，从中找到规律
		1	Machines can learn things by themselves 机器能自己学习东西
		0	Let the machine learn human thinking ability 让机器学习人的思想能力
ASAP-ZH	10	2	White: make the indoor temperature not too high, 白色使室内气温不太高 experiments show that white has the lowest light energy absorption rate 实验表明白色对光的能量吸收率最低
		1	Black allows the doghouse to absorb more heat in the light, making it warm 黑色能让狗窝在光下吸更多的热，使其温暖
		0	Dark gray: keep the temperature unchanged, 深灰色:: 使温度不变， the lighter the color, the lower the temperature 颜色越浅温度越低
ASAP-ZH <sup>MT</sup>	10	2	white :: having white paint would make the dog house colder, 白色:: 有白色油漆会使狗屋更冷， so in the summer the dog would not be hot. 所以在夏天狗不会很热。 The average for white is the coolest temperature ( 42 ( DEG ) ) 白色的平均值是最酷的 温度 (42 (DEG))
		1	black :: Because, the darker the lid color, 黑色:: 因为，盖子颜色越深， the greater the increase in the air temperature in the glass jar. 玻璃罐中空气温度的升高就越大。
		0	light gray :: The light grey will effect the doghouse by making it more noticeable 浅灰色：浅灰色会使狗狗更加显眼， and plus dogs can only see black, white and grey. 加上狗只能看到黑色，白色和灰色。

Table 5: Example answers in our data sets.

1- to 5-grams. Due to the limited amount of data, we use 10-fold cross-validation on both data sets.

For evaluation, we use accuracy, i.e., the percentage of student answers scored correctly, as well as QWK, which does not only consider whether an answer is classified correctly or not, but also how far it is from the gold standard classification.

## 5.2 Feature Sets

**Token Baseline** As a baseline, we follow previous work and use tokenization as segmentation, based on the HanLP tokenizer (He, 2020).

**Pinyin Features** In order to reduce the variance caused by spelling errors, we transcribe the text into Pinyin using *cnchar* (Chen, 2020) and extract ngrams on the level of transcribed characters. Note that we did not include information about tones in Pinyin

on purpose, in order to cover spelling errors caused by homophones.

**Character Features** For this segmentation level, we simply split a text into individual characters.

**Component Features** To extract these features on sub-character level, we use a dictionary with 17,803 Chinese characters<sup>7</sup> and their components to decompose all characters.

**Radical Features** Remember that radicals are only those components carrying the meaning of characters and might therefore be particularly useful in content scoring. We use XMNLP (Li, 2019) to extract the radicals of each character and use only those as features. Note that some radicals as defined by the “Table of Indexing Chinese Character Compo-

<sup>7</sup><https://github.com/kfcd/chaizi>

ID	Prompt	IAA (QWK)	avg. Length	Distribution		
				score 0	score 1	score 2
1	why we can use diamond cut glass 为什么 我们 能 用 钻石 切 玻璃?	.94	9.6	60%	38%	2%
2	why red clothes looks as red 为什么 红色 衣服 看起来 是 红色的?	.83	14.7	63%	20%	17%
3	what is artificial intelligence 什么 是 人工 智能 ?	.91	15.3	39%	37%	24%
4	what is natural language 什么 是 自然 语言 ?	.93	12.1	66%	15%	19%
5	what is machine learning 什么 是 机器 学习 ?	.89	15.7	31%	36%	33%

Table 6: Overview of prompts in CESA

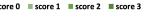
ID	IAA (QWK)	avg. Length	Distribution
			
1	.72	35.3	20% 28% 34% 18%
2	.70	38.2	36% 43% 18% 3%
10	.69	37.6	54% 32% 14%

Table 7: Overview of prompts in ASAP-ZH

ID	IAA (QWK)	avg. Length	Distribution
			
1	.96	68	22% 27% 31% 20%
2	.94	94	14% 25% 36% 25%
10	.91	61	17% 47% 36%

Table 8: Overview of prompts in ASAP-ZH<sup>MT</sup>

nents”<sup>8</sup> can consist of more than one component, therefore the radicals are not a proper subset of the components extracted above.

**Stroke Features** We use the *cnchar* tool to represent each answer as a sequence of individual strokes, following the stroke order for each character. Although we show the strokes in their original shapes in Table 9, a letter encoding is used in the experiment for an efficient processing.

**Auxiliary Words Removal** Based on the knowledge database released by Han et al. (2011), which contains 45 common auxiliary words in modern Chinese, we remove all these grammatical morphemes on token level to reduce the influence of expression variance. In our example shown in Table 9, the possessive

Answer	diamond 's hardness great 钻石 的 硬度 大
Tokens	钻石, 的, 硬度, 大
Pinyin	Zuan, Shi, De, Ying, Du, Da
Characters	钻, 石, 的, 硬, 度, 大
Components	金占, 一丿口, 白勺, 石更, 广廿又, 人一
Radicals	钅, 石, 白, 石, 广, 大
Strokes	丨一一丨丨一丨𠂊, 一丨丨𠂊, 丨丨𠂊—丨𠂊, 一丨丨𠂊—丨𠂊—丨𠂊, 丶—丨—丨丨—フ丨𠂊, 一丨𠂊

Table 9: Different segmentation levels for an answer in CESA, prompt 1.

‘s marker 的 is eliminated.

## 6 Results and Discussion

Table 10 shows the performance of the different system configurations for the individual data sets, per prompt as well as averaged over all prompts from the same data set. First, we see that all feature sets were able to learn something meaningful from the training data. Although the performance of different feature sets is quite close to each other, we see a slight but significant advantage across data sets of component and character features over the token baseline.

In order to check if tokenization caused

<sup>8</sup>[http://www.moe.gov.cn/s78/A19/yxs\\_left/moe\\_810/s230/201001/t20100115\\_75694.html](http://www.moe.gov.cn/s78/A19/yxs_left/moe_810/s230/201001/t20100115_75694.html)

Data Set	CESA						ASAP-ZH				ASAP-ZH <sup>MT</sup>			
	1	2	3	4	5	avg.	1	2	10	avg.	1	2	10	avg.
<b>Prompt</b>	.91	.84	.59	.66	.48	.70	.54	.40	.50	.48	.66	.59	.63	.63
<b>Token</b>	.02	.03	.03	.01	.03	.01**	.13	.01	.04	.09**	.02	.01	.01	.00
<b>Pinyin</b>	.01	.03	.00	.11	.05	.04**	.13	.03	.06	.07**	.00	.04	.04	.02*
<b>Character</b>	.03	.03	.01	.10	.02	.02**	.17	.04	.08	.10**	.01	.00	.04	.01**
<b>Component</b>	.02	.02	.03	.07	.00	.02**	.08	.08	.02	.06**	.02	.02	.04	.01
<b>Radical</b>	.01	.02	.02	.06	.04	.01**	.14	.07	.04	.08**	.01	.02	.03	.02**
<b>Stroke</b>	.01	.00	.00	.06	.04	.01**	.01	.00	.01	.01**	.01	.01	.01	.01**
<b>- Auxiliary</b>	.00	.00	.03	.02	.01	.01**	.00	.00	.00	.01**	.00	.01	.01	.01**

\*\*  $p < 0.01$ , \*  $p < 0.05$

Table 10: Classification results on different feature sets in QWK values.

problems in scoring, we manually inspected 100 answers from prompt 1 and 4 in CESA. However, we found that tokenization was only erroneous in 12 cases. Surprisingly, most of them occurred in prompt 1, where the token baseline even outperformed the character features and not in prompt 4, where character features performed better.

We also had a closer look at a number of student answers which are assigned a wrong score by the token baseline model but not by models with more fine-grained features. 7 out of 18 instances contain indeed variants of more frequent words in the data set. For example, human human  
人们 and 人 are less-frequently seen variants of 人类, all of which are indicators of a correct answer. This supports the assumption that, like in English, character-level features can capture variance in learner answers, in this case by handling variance in lexical choice.

The usage of Pinyin did not bring the expected benefit, possibly because the amount of spelling errors is not substantial enough in the data. Similarly, removing auxiliary words appears to have little influence on scoring performance.

## 7 Summary & Future Work

In this paper, we discussed the main challenges in Chinese content scoring in comparison with English, namely segmentation and a different form of linguistic variance. We reviewed related work in Chinese content scoring and saw a need for open-access scoring data sets in Chinese. Therefore, we collected two new data

sets, CESA and ASAP-ZH, and release them for research in the future.

While previous work has been limited to word-level features, we conducted a comparison of features on different segmentation levels. Although the difference between feature sets was in general small, we found that some answers with unusual expressions have a tendency to be better scored with models trained on lower level features, such as character n-grams.

In the future, we will extend our comparison of segmentation levels also to a deep learning setting, using embeddings of different granularity (Yin et al., 2016).

## References

- Hsuan-Chih Chen. 1992. Reading comprehension in chinese: Implications from character reading times. *Language processing in Chinese*, pages 175–205.
- Keh-Jiann Chen and Shing-Huan Liu. 1992. Word identification for mandarin chinese sentences. In *Proceedings of the 14th conference on Computational linguistics- Volume 1*, pages 101–107. Association for Computational Linguistics.
- Tack Chen. 2020. Full-featured, multi-end support for hanyu pinyin strokes js library.
- Zheng Chen and Kai-Fu Lee. 2000. A new statistical approach to chinese pinyin input. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 241–247.
- Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning*, 20(3):273–297.

- Johannes Daxenberger, Oliver Ferschke, Iryna Gurevych, and Torsten Zesch. 2014. Dkpro tc: A java-based framework for supervised learning experiments on textual data. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 61–66.
- Myroslava O Dzikovska, Rodney D Nielsen, Chris Brew, Claudia Leacock, Danilo Giampiccolo, Luisa Bentivogli, Peter Clark, Ido Dagan, and Hoa T Dang. 2013. Semeval-2013 task 7: The joint student response analysis and 8th recognizing textual entailment challenge. Technical report, NORTH TEXAS STATE UNIV DENTON.
- Ying-Jie Han, Hong-Ying Zan, Kun-Li Zhang, and Yu-Mei Chai. 2011. Automatic annotation of auxiliary words usage in rule-based chinese language. *Jisuanji Yingyong/Journal of Computer Applications*, 31(12):3271–3274.
- Han He. 2020. [HanLP: Han Language Processing](#).
- Michael Heilman and Nitin Madnani. 2013. Ets: Domain adaptation and stacking for short answer scoring. In *Second Joint Conference on Lexical and Computational Semantics (\* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 275–279.
- Andrea Horbach, Yuning Ding, and Torsten Zesch. 2017. The influence of spelling errors on content scoring performance. In *Proceedings of the 4th Workshop on Natural Language Processing Techniques for Educational Applications (NLPTEA 2017)*, pages 45–53.
- Andrea Horbach and Torsten Zesch. 2019. The influence of variance in learner answers on automatic content scoring. *Frontiers in Education*, 4:28.
- Chu-Ren Huang, Keh-Jiann Chen, and Li-Li Chang. 1996. Segmentation standard for chinese natural language processing. In *Proceedings of the 16th Conference on Computational Linguistics - Volume 2*, COLING ’96, page 1045–1048, USA. Association for Computational Linguistics.
- Yuwei Huang, Xi Yang, Fuzhen Zhuang, Lishan Zhang, and Shengquan Yu. 2018. Automatic chinese reading comprehension grading by lstm with knowledge adaptation. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 118–129. Springer.
- Nikolina Koleva, Andrea Horbach, Alexis Palmer, Simon Ostermann, and Manfred Pinkal. 2014. Paraphrase detection for short answer scoring. In *Proceedings of the third workshop on NLP for computer-assisted language learning*, pages 59–73.
- Claudia Leacock and Martin Chodorow. 2003. C-rater: Automated scoring of short-answer questions. *Computers and the Humanities*, 37(4):389–405.
- Dongjin Li, Tianyuan Liu, Wei Pan, Xiaoyue Liu, Yuqing Sun, and Feng Yuan. 2019. Grading chinese answers on specialty subjective questions. In *CCF Conference on Computer Supported Cooperative Work and Social Computing*, pages 670–682. Springer.
- HT Li. 1977. The history of chinese characters. *Taipei, Taiwan: Lian-Jian*.
- Shen Li, Zhe Zhao, Renfen Hu, Wensi Li, Tao Liu, and Xiaoyong Du. 2018. Analogical reasoning on chinese morphological and semantic relations. *arXiv preprint arXiv:1805.06504*.
- Xianming Li. 2019. A lightweight chinese natural language processing toolkit. <https://github.com/SeanLee97/xmnlp>.
- Roger Mitton. 1987. Spelling checkers, spelling correctors and the misspellings of poor spellers. *Information processing & management*, 23(5):495–505.
- Roger Mitton. 1996. *English spelling and the computer*. Longman Group.
- Michael Mohler and Rada Mihalcea. 2009. Text-to-text semantic similarity for automatic short answer grading. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 567–575. Association for Computational Linguistics.
- Jany Rademakers, Th J Ten Cate, and PR Bär. 2005. Progress testing with short answer questions. *Medical teacher*, 27(7):578–582.
- Brian Riordan, Andrea Horbach, Aoife Cahill, Torsten Zesch, and Chungmin Lee. 2017. Investigating neural architectures for short answer scoring. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 159–168.
- Keisuke Sakaguchi, Michael Heilman, and Nitin Madnani. 2015. Effective feature integration for automated short answer scoring. In *Proceedings of the 2015 conference of the North American Chapter of the association for computational linguistics: Human language technologies*, pages 1049–1054.
- Hanqing Tao, Shiwei Tong, Tong Xu, Qi Liu, and Enhong Chen. 2019. Chinese embedding via stroke and glyph information: A dual-channel view. *arXiv preprint arXiv:1906.04287*.
- Hao-Chuan Wang, Chun-Yen Chang, and Tsai-Yen Li. 2008. Assessing creative problem-solving with automated text grading. *Computers & Education*, 51(4):1450–1466.

- Ian H Witten, Eibe Frank, Leonard E Trigg, Mark A Hall, Geoffrey Holmes, and Sally Jo Cunningham. 1999. Weka: Practical machine learning tools and techniques with java implementations.
- Shih-Hung Wu and Wen-Feng Shih. 2018. A short answer grading system in chinese by support vector approach. In *Proceedings of the 5th Workshop on Natural Language Processing Techniques for Educational Applications*, pages 125–129.
- Shih-Hung Wu and Chun-Yu Yeh. 2019. A short answer grading system in chinese by cnn. In *2019 IEEE 10th International Conference on Awareness Science and Technology (iCAST)*, pages 1–5. IEEE.
- Shaohua Yang, Hai Zhao, Xiaolin Wang, and Bao-liang Lu. 2012. Spell checking for chinese. In *LREC*, pages 730–736.
- Xi Yang, Yuwei Huang, Fuzhen Zhuang, Lishan Zhang, and Shengquan Yu. 2018. Automatic Chinese Short Answer Grading with Deep Autoencoder. In *Artificial Intelligence in Education*, pages 399–404, Cham. Springer International Publishing.
- Xi Yang, Lishan Zhang, and Shengquan Yu. 2017. Can short answers to open response questions be auto-graded without a grading rubric? In *International Conference on Artificial Intelligence in Education*, pages 594–597. Springer.
- Rongchao Yin, Quan Wang, Peng Li, Rui Li, and Bin Wang. 2016. Multi-granularity chinese word embedding. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 981–986.
- Hongying Zan and Xuefeng Zhu. 2009. Nlp oriented studies on chinese functional words and the construction of their generalized knowledge base. *Contemporary Linguistics*, 2:124–135.
- Torsten Zesch, Michael Heilman, and Aoife Cahill. 2015. Reducing annotation efforts in supervised short answer scoring. In *Proceedings of the Tenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 124–132.
- Torsten Zesch and Andrea Horbach. 2018. Escrito-an nlp-enhanced educational scoring toolkit. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*.
- Hai Zhao, Deng Cai, Changning Huang, and Chunyu Kit. 2019. Chinese word segmentation: Another decade review (2007-2017). *arXiv preprint arXiv:1901.06079*.
- Ramon Ziai, Niels Ott, and Detmar Meurers. 2012. Short answer assessment: Establishing links between research strands. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, pages 190–200. Association for Computational Linguistics.

# Analysis of Hierarchical Multi-Content Text Classification Model on B-SHARP Dataset for Early Detection of Alzheimer’s Disease

<b>Renxuan A. Li</b> Computer Science Emory University Atlanta, GA, USA <code>albert.li@emory.edu</code>	<b>Ihab Hajjar</b> Neurology Emory University Atlanta, GA, USA <code>ihajjar@emory.edu</code>	<b>Felicia Goldstein</b> Neurology Emory University Atlanta, GA, USA <code>fgoldst@emory.edu</code>	<b>Jinho D. Choi</b> Computer Science Emory University Atlanta, GA, USA <code>jchoi31@emory.edu</code>
--	---	---	--

## Abstract

This paper presents a new dataset, B-SHARP, that can be used to develop NLP models for the detection of Mild Cognitive Impairment (MCI) known as an early sign of Alzheimer’s disease. Our dataset contains 1-2 min speech segments from 326 human subjects for 3 topics, (1) daily activity, (2) room environment, and (3) picture description, and their transcripts so that a total of 650 speech segments are collected. Given the B-SHARP dataset, several hierarchical text classification models are developed that jointly learn combinatory features across all 3 topics. The best performance of 74.1% is achieved by an ensemble model that adapts 3 types of transformer encoders. To the best of our knowledge, this is the first work that builds deep learning-based text classification models on multiple contents for the detection of MCI.

## 1 Introduction

Alzheimer’s Disease (AD) is a progressive neurodegenerative disorder that is associated with memory loss and declines in major brain functions including semantic and pragmatic levels of language processing (Vestal et al., 2006; Ferris and Farlow, 2013). Traditional cognitive assessments such as positron emission tomography or cerebrospinal fluid analysis are expensive and time-consuming (Fyffe et al., 2011). This may cause delay in treating AD, known to be irreversible and incurable (Korczyn, 2012), and put an increasing pressure on public health, especially for seniors whose life expectancy is rapidly growing yet are more likely to develop AD. Thus, it is crucial to find a more intelligent way of detecting AD in the earliest stage possible (Karr et al., 2018).

Mild Cognitive Impairment (MCI) is considered the first phase that patients start having biomarker evidence of brain changes that can eventually lead to AD (Albert et al., 2011). MCI involves subtle language changes from impairment in reasoning

that may not be noticeable to people other than friends and relatives. Because of this, the detection of MCI is a much more challenging task than detecting dementia (Suzman and Beard, 2011). Recent studies in NLP have shown that it is possible to detect early stages of AD by analyzing patients’ language patterns; however, most previous works have focused on the detection of dementia instead and researches tackling the detection of MCI have been based on relatively small datasets (Section 2).

This paper presents a new dataset that comprises three types of speech segments from both normal controls and MCI patients (Section 3). Then, a hierarchical text classification model is proposed, which jointly learns features from all three types of speech segments to determine whether or not each subject has MCI (Section 4). Individual and ensemble models using three types of transformer encoders are evaluated on our dataset and show that different transformer encoders reveal strengths in distinct types of speeches (Section 5). We believe that this work takes the initiative of deep learning-based NLP for detecting MCI that will be broadly beneficial to global public health.

## 2 Related Work

Only few studies have tackled the detection of MCI using NLP.<sup>1</sup> Asgari et al. (2017) conducted interviews with (27<sub>C</sub>, 14<sub>M</sub>), and developed SVM and random forest models on their transcribed speeches. Beltrami et al. (2018) conducted three speech tasks with (48<sub>C</sub>, 32<sub>M</sub>, 16<sub>D</sub>), and analyzed phonetic and linguistic features of their speeches and transcripts. Fraser et al. (2019) conducted 3 language tasks with (29<sub>C</sub>, 26<sub>M</sub>), and built a cascade model to learn multimodal features such as audio, text, eye-tracking. Gosztolya et al. (2019) conducted question answer-

<sup>1</sup>#<sub>C</sub>: the number of normal controls,  
#<sub>M/D/A</sub>: the number of MCI / Dementia / AD patients.

		Tokens	Sentences	Nouns	Verbs	Conjuncts	Complex	Discourse
<b>Q<sub>1</sub></b>	Control	186.6 ( $\pm 60.4$ )	10.4 ( $\pm 4.5$ )	28.1 ( $\pm 9.6$ )	30.4 ( $\pm 11.5$ )	8.5 ( $\pm 4.5$ )	2.3 ( $\pm 1.7$ )	8.1 ( $\pm 5.4$ )
	MCI	175.6 ( $\pm 54.5$ )	9.8 ( $\pm 4.1$ )	23.7 ( $\pm 8.3$ )	29.3 ( $\pm 10.4$ )	8.5 ( $\pm 4.2$ )	2.0 ( $\pm 1.6$ )	9.2 ( $\pm 6.0$ )
<b>Q<sub>2</sub></b>	Control	191.5 ( $\pm 11.8$ )	11.7 ( $\pm 4.7$ )	41.1 ( $\pm 13.3$ )	24.3 ( $\pm 11.2$ )	6.6 ( $\pm 4.5$ )	3.6 ( $\pm 2.7$ )	7.1 ( $\pm 4.8$ )
	MCI	178.6 ( $\pm 11.7$ )	11.6 ( $\pm 4.7$ )	36.7 ( $\pm 12.1$ )	23.2 ( $\pm 10.6$ )	6.4 ( $\pm 4.4$ )	2.9 ( $\pm 2.3$ )	8.4 ( $\pm 5.3$ )
<b>Q<sub>3</sub></b>	Control	193.4 ( $\pm 63.4$ )	12.6 ( $\pm 5.4$ )	39.5 ( $\pm 13.5$ )	28.4 ( $\pm 10.1$ )	8.0 ( $\pm 4.8$ )	3.3 ( $\pm 2.1$ )	6.1 ( $\pm 5.5$ )
	MCI	187.8 ( $\pm 63.4$ )	12.7 ( $\pm 5.1$ )	36.2 ( $\pm 13.2$ )	27.7 ( $\pm 10.9$ )	7.2 ( $\pm 4.2$ )	2.6 ( $\pm 2.0$ )	7.3 ( $\pm 5.5$ )
<b>All</b>	Control	<b>578.1</b> ( $\pm 149.8$ )	34.5 ( $\pm 10.7$ )	<b>110.5</b> ( $\pm 27.9$ )	84.2 ( $\pm 25.4$ )	23.5 ( $\pm 10.1$ )	<b>9.3</b> ( $\pm 4.5$ )	21.4 ( $\pm 13.0$ )
	MCI	548.7 ( $\pm 140.6$ )	34.0 ( $\pm 10.5$ )	98.1 ( $\pm 26.1$ )	81.2 ( $\pm 24.1$ )	22.5 ( $\pm 9.7$ )	7.7 ( $\pm 4.2$ )	<b>25.3</b> ( $\pm 15.0$ )
	<i>p</i>	<b>0.0110</b>	0.5541	< 0.0001	0.1277	0.2046	< 0.0001	<b>0.0006</b>

Table 1: Average counts and their standard deviations of linguistic features per transcript in the B-SHARP dataset. Complex: occurrences of complex structures (e.g., relative clauses, non-finite clauses), Discourse: occurrences of discourse elements (e.g., interjections, disfluency).

ing sessions with (25<sub>C</sub>, 25<sub>M</sub>, 25<sub>A</sub>), and trained a SVM model using acoustic and linguistic features. All of the previous works were based on fewer than 100 subjects using traditional linguistic features to develop NLP models, compared to our work that is based on 326 subjects and 650 recordings using the latest transformer-based deep neural models.

The task of dementia detection has been more explored by the NLP community. Becker et al. (1994) presented the DementiaBank, that consists of 552 audio recordings describing the picture called “*The Boston Cookie Theft*” from 99 normal controls and 194 dementia patients, that have been used by the following works. Orimaye et al. (2016) presented deep-deep neural network language models using higher-order  $n$ -grams and skip-grams. Pou-Prom and Rudzicz (2018) leveraged linguistic features and multiview embeddings by applying generalized canonical correlation analysis. Karleka et al. (2018) proposed a model based on convolutional and recurrent neural networks and gave interpretations of this model to explain linguistic characteristics for detecting dementia. Our work is distinguished as:

- We tackle the detection of MCI, not dementia,
- Our documents are multi-contents compared to single-content documents in the DementiaBank.
- Our approach is based on the latest contextualized embeddings compared to the distributional embeddings adapted by the previous works.

### 3 Dataset

#### 3.1 B-SHARP

Our work is based on data collected as part of the Brain, Stress, Hypertension, and Aging Research Program (B-SHARP).<sup>2</sup> In this dataset, 185 normal

controls and 141 MCI patients are selected based on neuropsychological and clinical assessments. Every subject has been examined with multiple cognitive tests including the Montreal Cognitive Assessment (MoCA; Nasreddine et al. 2005) and the Boston Naming Test (BNT; Kaplan et al. 1983), followed by a speech task protocol for recording. 51.5% and 23.9% of the subjects have so far come back for their 2nd and 3rd visits to take new voice recordings, respectively. B-SHARP is an ongoing program; recordings of 20-25 subjects are taken every month; thus, the data is still growing.

	Sbj	2nd	3rd	Rec	MoCA	BNT
C	185	100	50	385	26.2 ( $\pm 2.6$ )	14.2 ( $\pm 1.2$ )
M	141	68	28	265	21.5 ( $\pm 3.5$ )	13.4 ( $\pm 1.5$ )
$\Sigma$	326	168	78	650	24.2 ( $\pm 3.8$ )	13.9 ( $\pm 1.4$ )

Table 2: Statistics of control (C) and MCI (M) groups. Sbj: # of subjects, 2nd/3rd: # of subjects who made the 2nd/3rd visits, Rec: # of voice recordings, MoCA/BNT: average scores and stdevs from MoCA/BNT. Note that subjects with the 2nd/3rd visits take one/two additional recordings; thus, Rec = Sbj + 1·(2nd) + 2·(3rd).

Table 2 shows the statistics of the control and the MCI groups in B-SHARP. Note that when subjects make multiple visits, there is a year gap in between so that subjects generally do not remember so much from their previous visits. Thus, speeches from the same subject are not necessarily more similar than ones from the other subjects. In fact, most speeches across subjects, regardless of their groups, are very similar when they are transcribed since all subjects follow the same speech protocol in Section 3.2.<sup>3</sup>

#### 3.2 Speech Task Protocol

A speech task protocol has been conducted to collect recordings of both control and MCI subjects

<sup>2</sup>B-SHARP: <http://medicine.emory.edu/bsharp>

<sup>3</sup>A.3 compares B-SHARP with the DementiaBank in details.

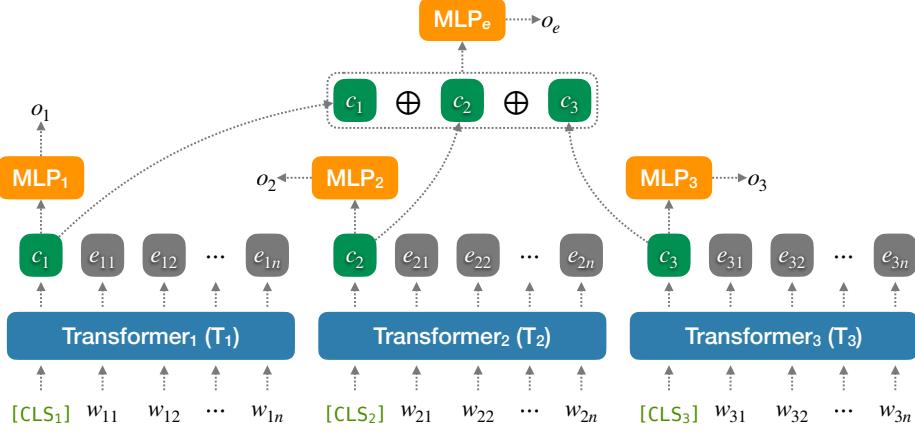


Figure 1: Overview of hierarchical transformer to combine content features from the three types of speech tasks.

who are asked to speak about  $Q_1$ : daily activity,  $Q_2$ : room environment, and  $Q_3$ : picture description for 1-2 minutes each. All subjects are provided with the same instructions in A.2, and visual abilities of the subjects are confirmed before recording. To reduce potential variance, the subjects are guided to follow similar activities before  $Q_1$ , located to similar room settings before  $Q_2$ , and shown the same picture in Fig 2, “*The Circus Procession*”, for  $Q_3$ .

The collected voice recordings are automatically transcribed by the online tool called Temi.<sup>4</sup> Table 1 shows linguistic features about our dataset analyzed by the open-source NLP toolkit, ELIT.<sup>5</sup> Transcripts from the control group depict significantly higher numbers of tokens, nouns, and complex structures while transcripts from the MCI group gives significantly more discourse elements, implying that the control subjects are more expressive while the MCI subjects include more disfluency in their speeches.

#### 4 Hierarchical Transformer

Although transformer encoders have recently established the state-of-the-art results on most document classification tasks, they have a limit on the input size. As in Table 1, the average number of tokens in our input documents well-exceeds 512 when combining transcripts from all three tasks, which is the max-number of tokens that the pretrained models of these transformers allow in general. This makes it difficult to simply join all transcripts together and feed into a transformer encoder. Thus, this section presents a hierarchical transformer to overcome the challenge of long documents while jointly training transcript contents from all three tasks (Figure 1).

Let  $W_i = \{w_{i1}, \dots, w_{in}\}$  be a transcript, where  $w_{ij}$  represents the  $j$ 'th token in the transcript produced by the  $i$ 'th task  $Q_i$  (in our case,  $i = \{1, 2, 3\}$ ).  $W_i$  is prepended by the special token  $[\text{CLS}_i]$  that is used to learn the transcript embedding, and fed into the transformer  $T_i$ . The transformer then generates  $E_i = \{c_i, e_{i1}, \dots, e_{in}\}$ , where  $c_i$  and  $e_{ij}$  are the embeddings for  $[\text{CLS}_i]$  and  $w_{ij}$ , respectively.  $c_i \in \mathbb{R}^d$  is used to make two types of predictions.

First,  $c_i$  is fed into a multilayer perceptron layer,  $\text{MLP}_i$ , that generates the output vector  $o_i \in \mathbb{R}^2$  to predict whether or not the subject has MCI based on the transcript from  $Q_i$  alone. Second, the transcript embeddings from all three tasks are concatenated such that  $c_e = c_1 \oplus c_2 \oplus c_3 \in \mathbb{R}^{3d}$ , which gets fed into another  $\text{MLP}_e$  to generate the output vector  $o_e \in \mathbb{R}^2$ , and makes the binary decision based on the transcripts from all three tasks,  $Q_1$ ,  $Q_2$  and  $Q_3$ .

#### 5 Experiments

There are 650 recordings in our dataset (Table 2), that is rather small to divide into train, development, and test sets. Thus, 5-fold cross-validation (CV) is used to evaluate the performance of our models. Table 5 shows the distributions of the five CV sets for our experiments, where the transcript of each recording is treated as an independent document. Notice that the distributions are calculated based on analysis of the last MLP layer instead of simple majority vote on individual models.

It is worth mentioning that all recordings from the same subject given multiple visits are assigned to the same CV set; thus, there is no overlap in terms of subjects across these CV sets. This allows us to avoid potential inflation in accuracy due to unique language patterns used by individual subjects.

<sup>4</sup>Temi: <https://www.temi.com>

<sup>5</sup>ELIT: <https://github.com/elitcloud/elit>

	BERT			RoBERTa			ALBERT		
	Q <sub>1</sub>	Q <sub>2</sub>	Q <sub>3</sub>	Q <sub>1</sub>	Q <sub>2</sub>	Q <sub>3</sub>	Q <sub>1</sub>	Q <sub>2</sub>	Q <sub>3</sub>
ACC	67.6 ( $\pm 0.4$ )	<b>69.0</b> ( $\pm 1.2$ )	67.7 ( $\pm 0.7$ )	69.0 ( $\pm 1.5$ )	<b>69.9</b> ( $\pm 0.2$ )	65.2 ( $\pm 0.3$ )	67.6 ( $\pm 1.5$ )	<b>69.5</b> ( $\pm 0.3$ )	66.6 ( $\pm 1.3$ )
SEN	48.9 ( $\pm 1.8$ )	<b>57.1</b> ( $\pm 2.5$ )	41.5 ( $\pm 3.6$ )	44.3 ( $\pm 4.5$ )	<b>55.3</b> ( $\pm 1.2$ )	37.1 ( $\pm 3.7$ )	45.9 ( $\pm 1.9$ )	<b>52.2</b> ( $\pm 0.6$ )	37.4 ( $\pm 3.3$ )
SPE	80.4 ( $\pm 1.2$ )	77.3 ( $\pm 2.8$ )	<b>85.2</b> ( $\pm 3.0$ )	<b>85.8</b> ( $\pm 2.1$ )	79.7 ( $\pm 0.7$ )	84.5 ( $\pm 3.0$ )	82.6 ( $\pm 3.7$ )	81.4 ( $\pm 0.3$ )	<b>86.8</b> ( $\pm 3.3$ )

Table 3: Model performance on the individual tasks. ACC: accuracy, SEN: sensitivity, SPE: specificity.

	CNN	BERT <sub>e</sub>	RoBERTa <sub>e</sub>	ALBERT <sub>e</sub>	B <sub>e</sub> + R <sub>e</sub>	A <sub>e</sub> + R <sub>e</sub>	B <sub>e</sub> + A <sub>e</sub> + R <sub>e</sub>
ACC	69.5 ( $\pm 0.2$ )	69.9 ( $\pm 1.1$ )	71.6 ( $\pm 1.5$ )	69.7 ( $\pm 2.9$ )	72.2 ( $\pm 0.7$ )	71.5 ( $\pm 1.9$ )	<b>74.1</b> ( $\pm 0.3$ )
SEN	49.2 ( $\pm 0.8$ )	57.6 ( $\pm 3.4$ )	48.5 ( $\pm 6.1$ )	46.2 ( $\pm 8.3$ )	56.5 ( $\pm 2.5$ )	51.7 ( $\pm 1.3$ )	<b>60.9</b> ( $\pm 5.2$ )
SPE	83.5 ( $\pm 0.9$ )	77.4 ( $\pm 4.8$ )	<b>87.5</b> ( $\pm 1.8$ )	85.4 ( $\pm 0.5$ )	83.1 ( $\pm 0.9$ )	86.7 ( $\pm 3.4$ )	84.0 ( $\pm 2.4$ )

Table 4: Performance of ensemble models. Bert<sub>e</sub>/RoBERTa<sub>e</sub>/ALBERT<sub>e</sub> use transcript embeddings from all 3 tasks trained by the BERT/RoBERTa/ALBERT models in Table 3, respectively. B<sub>e</sub>+R<sub>e</sub> uses transcript embeddings from both Bert<sub>e</sub> and RoBERTa<sub>e</sub> (so the total of 6 embeddings), A<sub>e</sub>+R<sub>e</sub> uses transcript embeddings from both ALBERT<sub>e</sub> and RoBERTa<sub>e</sub> (6 embeddings), and B<sub>e</sub>+A<sub>e</sub>+R<sub>e</sub> uses transcript embeddings from all three models (9 embeddings).

Three transformer encoders are used, BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2020), and ALBERT (Lan et al., 2019) for our experiments. Every model is trained 3 times and its average performance with the standard deviation are reported.<sup>6</sup>

	CV <sub>0</sub>	CV <sub>1</sub>	CV <sub>2</sub>	CV <sub>3</sub>	CV <sub>4</sub>	ALL
C <sub>Rec</sub>	77	77	77	77	77	<b>385</b>
M <sub>Rec</sub>	53	53	53	53	53	<b>265</b>
C <sub>Sbj</sub>	37	37	37	37	37	<b>185</b>
M <sub>Sbj</sub>	27	28	28	29	29	<b>141</b>

Table 5: Statistics of the CV sets for our experiments. Rec/Sbj: # of recordings/subjects, C/M: in control/MCI group. CV<sub>i</sub>: the  $i$ 'th set. ALL:  $\sum_{i=0}^4 \text{CV}_i$ .

## 5.1 Performance of Individual Models

Individual models are built by training transcripts from each task separately using MLP<sub>i</sub> in Section 4. Table 3 shows the performance of the 3 transformer models on the individual tasks. The performance on Q<sub>2</sub> shows the highest accuracy for all three models, achieving 69.9% with RoBERTa, implying that the room environment task of Q<sub>2</sub>, involving many spatial descriptions, are the most effective to distinguish the MCI group. The highest sensitivity of 57.1% is achieved by BERT on Q<sub>2</sub>, and the highest specificity of 86.8% is achieved by ALBERT on Q<sub>3</sub>. Such a low sensitivity and a high specificity imply that it is easier to recognize the normal controls but not the MCI patients given the short speeches.

## 5.2 Performance of Ensemble Models

Ensemble models are developed by jointly training multiple transcript embeddings from the individual models using MLP<sub>e</sub> in Section 4. Table 4 shows the

<sup>6</sup>Details about the experimental settings are provided in A.1.

model performance of the ensemble models. Additionally, results from a model that takes transcripts from the 3 tasks as one input document and trains a convolutional neural network (CNN) are provided for comparison to Karleka et al. (2018).<sup>7</sup> R<sub>e</sub> shows 1.7% improvement on accuracy over the RoBERTa model in Table 3 although its sensitivity is worse. Table 6 shows the voting distributions of each task combination; given the samples correctly predicted by RoBERTa<sub>e</sub>, we count how often the individual models are correct for those samples by comparing the weights in MLP<sub>e</sub> and estimate the percentages. The combination of (Q<sub>1</sub>, Q<sub>3</sub>) shows the highest percentage of 30%, meaning that 30% of the corrected predicted samples are voted by both Q<sub>1</sub> and Q<sub>3</sub>.

Q <sub>1</sub>	Q <sub>2</sub>	Q <sub>3</sub>	Q <sub>1,2</sub>	Q <sub>1,3</sub>	Q <sub>2,3</sub>	Q <sub>1,2,3</sub>
5.8	6.4	2.8	19.5	<b>30.0</b>	8.8	26.1

Table 6: Voting distributions of each task combination for RoBERTa<sub>e</sub>. Q<sub>i</sub>: % of only the Q<sub>i</sub> model is correct, Q<sub>i,i,j</sub>: % of all Q<sub>i</sub>, Q<sub>i</sub>, and Q<sub>j</sub> models are correct.

A similar analysis is done for B<sub>e</sub>+R<sub>e</sub>+A<sub>e</sub> although displaying the distributions is quite infeasible since it involves 2<sup>9</sup>-1 combinations. Among the samples correctly predicted by B<sub>e</sub>+R<sub>e</sub>+A<sub>e</sub>, 86% are derived from majority votes; in other words, at least 5 out of 9 individual models agree with the predictions. Votes from 6 and 5 models are the largest groups, showing 35% and 28%, respectively. Only 0.21% are agreed by all 9 models. No case of votes from 3 or less models is found, implying that no individual model dominates the final decision of B<sub>e</sub>+R<sub>e</sub>+A<sub>e</sub>.

<sup>7</sup>We also experimented with LSTM-RNN and CNN-LSTM models as suggested by Karleka et al. (2018); however, the CNN model gave the highest accuracy on our dataset.

## 6 Conclusion

This paper presents the B-SHARP dataset, that is the largest dataset for the task of MCI detection feasible to develop robust deep neural models. Our best ensemble model using hierarchical transformer gives the accuracy of 74% to distinguish MCI patients from normal controls that is very promising. We will also explore models to make a longevity analysis per patient with this dataset.<sup>8</sup>

## References

- Marilyn S Albert, Steven T DeKosky, Dennis Dickson, Bruno Dubois, Howard H Feldman, Nick C Fox, Anthony Gamst, David M Holtzman, William J Jagust, Ronald C Petersen, Peter J Snyder, Maria C Carrillo, Bill Thies, and Creighton H Phelps. 2011. *The Diagnosis of Mild Cognitive Impairment Due to Alzheimer’s Disease: Recommendations From the National Institute on Aging-Alzheimer’s Association Workgroups on Diagnostic Guidelines for Alzheimer’s Disease.* *Alzheimer’s Dementia*, 7(3):270–279.
- Meysam Asgari, Jeffrey Kaye, and Hiroko Dodge. 2017. *Predicting Mild Cognitive Impairment From Spontaneous Spoken Utterances.* *Alzheimer’s Dementia*, 3(2):219–228.
- James T. Becker, Francois Boller, Oscar L. Lopez, Judith Saxton, and Karen L. McGonigle. 1994. *The natural history of alzheimer’s disease: description of study cohort and accuracy of diagnosis.* *Archives of Neurology*, 51(6):585–594.
- Daniela Beltrami, Gloria Gagliardi, Rema Rossini Favretti, Enrico Ghidoni, Fabio Tamburini, and Laura Calzà. 2018. *Speech Analysis by Natural Language Processing Techniques: A Possible Tool for Very Early Detection of Cognitive Decline?* *Frontiers in Aging Neuroscience*, 10(369):1–13.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.* In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186.
- Steven H Ferris and Martin Farlow. 2013. *Language Impairment in Alzheimer’s Disease and Benefits of Acetylcholinesterase Inhibitors.* *Clinical Interventions in Aging*, 8:1007–1014.
- Kathleen C. Fraser, Kristina Lundholm Fors, Marie Eckerström, Fredrik Öhman, and Dimitrios Kokkinakis. 2019. *Predicting MCI Status From Multi-modal Language Data Using Cascaded Classifiers.* *Frontiers in Aging Neuroscience*, 11(205):1–18.
- Denise C. Fyffe, Shubhabrata Mukherjee, Lisa L. Barnes, Jennifer J. Manly, David A. Bennett, and Paul K. Crane. 2011. *Explaining Differences in Episodic Memory Performance among Older African Americans and Whites: The Roles of Factors Related to Cognitive Reserve and Test Bias.* *Journal of the International Neuropsychological Society*, 17(4):625–638.
- Gábor Gosztolya, Veronika Vinczea, László Tóth, Magdolna Pákáskid, János Kálmand, and Ildikó Hoffmann. 2019. *Identifying Mild Cognitive Impairment and mild Alzheimer’s disease based on spontaneous speech using ASR and linguistic features.* *Computer Speech & Language*, 53:181–197.
- Edith Kaplan, Harold Goodglass, and Sandra Weintraub. 1983. *The Boston Naming Test.* Philadelphia: Lea and Febiger.
- Sweta Karleka, Tong Niu, and Mohit Bansal. 2018. *Detecting linguistic characteristics of alzheimer’s dementia by interpreting neural models.* In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 701–707, New Orleans, Louisiana. Association for Computational Linguistics.
- Justin E Karr, Raquel B Graham, Scott M Hofer, and Graciela Muniz-Terrera. 2018. *When Does Cognitive Decline Begin? A Systematic Review of Change Point Studies on Accelerated Decline in Cognitive and Neurological Outcomes Preceding Mild Cognitive Impairment, Dementia, and Death.* *Psychology and Aging*, 33(2):195–218.
- Amos D Korczyn. 2012. *Why Have We Failed to Cure Alzheimer’s Disease?* *Journal of Alzheimer’s Disease*, 29(2):275–282.
- Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. *ALBERT: A Lite BERT for Self-supervised Learning of Language Representations.* *arXiv*, 11942(1909).
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2020. *RoBERTa: A Robustly Optimized BERT Pretraining Approach.* *arXiv*, 1907(11692).
- Ziad S. Nasreddine, Natalie A. Phillips, Valérie Bédirian, Simon Charbonneau, Victor Whitehead, Isabelle Collin, Jeffrey L. Cummings, and Howard Chertkow. 2005. *The Montreal Cognitive Assessment, MoCA: a brief screening tool for mild cognitive impairment.* *Journal of the American Geriatrics Society*, 53(4):695–699.
- Sylvester Olubolu Orimaye, Jojo Sze-Meng Wong, and Judyanne Sharmini Gilbert Fernandez. 2016. *Deep-Deep Neural Network Language Models for Predicting Mild Cognitive Impairment.* In *Proceedings of*

<sup>8</sup>All our resources including source codes and models are available at <http://anonymous>.

*the IJCAI Workshop on Advances in Bioinformatics and Artificial Intelligence*, pages 14–20.

Chloé Pou-Prom and Frank Rudzicz. 2018. Learning multiview embeddings for assessing dementia. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP’18, pages 2812–2817, Brussels, Belgium. Association for Computational Linguistics.

Richard Suzman and John Beard. 2011. Global health and aging.

Lindsey Vestal, Laura Smith-Olinde, Gretchen Hicks, Terri Hutton, and John Hart Jr. 2006. Efficacy of Language Assessment in Alzheimer’s Disease: Comparing In-Person Examination and Telemedicine. *Clinical Interventions in Aging*, 1(4):467–471.

## A Appendix

### A.1 Experimental Settings

Table 7 shows the configuration of the transformer encoders in Section 5. The base pre-trained models are used for all encoders.

Transformer	L	AH	IC	HC	P
BERT	12	12	768	768	108M
RoBERTa	12	12	768	768	125M
ALBERT	12	12	768	128	12M

Table 7: Configurations of the BERT, RoBERTa, and ALBERT encoders for our experiments. L: # of layers, AH: # of attended heads, IC: # of input cells, HC: # of hidden cells, P: # of parameters.

**Individual Models** For training the BERT and RoBERTa models, the batch size of 5, the learning rate of  $5 \cdot 10^{-6}$ , and the gradient clip of norm 0.5 are used with the Adam optimizer. A dropout rate of 0.15 is applied to all layers. For the ALBERT model, the batch size of 8 is used. All three models are trained for 30 epochs.

**Ensemble Models** For training the two model ensembles,  $B_e+R_e$  and  $A_e+R_e$ , the batch size of 72 and the learning rate of  $5 \cdot 10^{-5}$  are used with the Adam optimizer for 200 epochs. A dropout rate of 0.25 is also applied. For training the  $B_e+A_e+R_e$  model, the dropout rate is set to 0.3.

### A.2 Speech Task Protocol

Table 8 describes the instructions provided to the subjects for the three speech tasks in Section 3.2.

Task	Instruction
$Q_1$	I would like you to describe to me everything we did from the moment we met today until now. Please try to recall as many details as possible in the order the events actually happened where we met, what we did, what we saw, where we went, and what you felt or thought during each of these events.
$Q_2$	I would like you to describe everything that you see in this room.
$Q_3$	I am going to show you a picture and ask you to describe what you see in as much detail as possible. You can describe the activities, characters, and colors of things you see in this picture.

Table 8: Instructions of the 3 speech tasks,  $Q_1$ ,  $Q_2$ ,  $Q_3$ , provided to the subjects.

Figure 2 illustrates the image of the picture called “The Circus Procession” for the picture description task,  $Q_3$ , copyrighted by the *McLoughlin Brothers* as part of the Juvenile Collection.



Figure 2: The picture of “The Circus Procession” used in the B-SHARP dataset.

### A.3 B-SHARP Compared to DementiaBank

DementiaBank is the largest public dataset for dementia detection that comprises recordings for 4 language tasks, picture description, verbal fluency, story recall, and sentence construction, from a large longitudinal study (Becker et al., 1994). Subjects in this study are divided into two groups, normal controls and dementia patients. Among the four tasks, data from only the picture description task can be used for classification since the other tasks give data of dementia patients only.<sup>9</sup> The design of this task is similar to  $Q_3$  in B-SHARP (Section 3.2); each subject is shown “The Boston Cookie Theft” picture in Figure 3 to describe for 1-2 minutes.

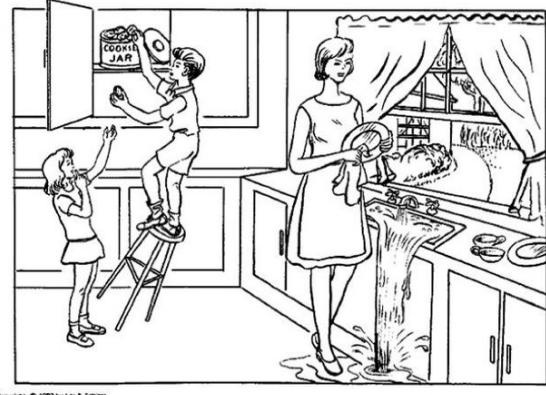


Figure 3: The picture of “The Boston Cookie Theft” used in the DementiaBank.

Table 10 shows the statistics of the DementiaBank in comparison to Table 2 in Section 3. Subjects in this study made up to 5 visits compared to 3 in B-SHARP although the number of subjects in each visit is larger in B-SHARP. B-SHARP has  $\approx 100$

<sup>9</sup>The verbal fluency task gives 1 audio recording of a normal control, that is still not enough to train classification models.

	Tokens	Sentences	Nouns	Verbs	Conjuncts	Complex	Discourse
Control	124.0 ( $\pm 59.7$ )	12.6 ( $\pm 5.1$ )	<b>23.7</b> ( $\pm 11.8$ )	<b>27.1</b> ( $\pm 11.9$ )	2.8 ( $\pm 2.8$ )	1.6 ( $\pm 1.6$ )	1.5 ( $\pm 1.6$ )
Dementia	114.3 ( $\pm 61.3$ )	12.1 ( $\pm 6.4$ )	18.7 ( $\pm 10.4$ )	23.9 ( $\pm 12.9$ )	2.4 ( $\pm 2.4$ )	1.4 ( $\pm 1.4$ )	<b>2.8</b> ( $\pm 2.9$ )
<i>p</i>	0.0625	0.3204	< 0.0001	<b>0.0029</b>	0.0715	0.1184	< 0.0001

Table 9: Average counts and standard deviations of linguistic features per transcript in the DementiaBank. See the caption in Table 1 for the column descriptions.

more recordings than the DementiaBank, more importantly, B-SHARP is still growing, which makes it the largest dataset for NLP research related to the detection of Alzheimer’s Disease. Unlike DementiaBank where 66.2% of the subjects are dementia patients, 43.3% of the subjects belong to the MCI group in B-SHARP; this makes sense because MCI is closer to the preclinical phase that involves a much fewer number of patients reported in general.

Group	Sbj	2nd	3rd	4th	5th	Rec
Control	99	29	28	9	8	243
Dementia	194	53	13	8	3	309
All	293	82	41	17	11	552

Table 10: Statistics of the control and the dementia groups in the DementiaBank. Sbj: # of subjects,  $i$ ’th: # of subjects who made  $i$ ’th visits, Rec: # of voice recordings. Note that subject with  $i$ ’th visits take  $(i - 1)$  additional recordings; thus,  $\text{Rec} = \text{Sbj} + \sum_{i=2}^5 (i - 1)$ ’th.

Table 9 shows the statistics of linguistic features in comparison to Table 1 in Section 3. The same tools, Temi and ELIT, are used to measure them. Unlike B-SHARP, the control group in the DementiaBank does not reveal a significantly greater number of tokens than the dementia group. The document size in the DementiaBank is 4.9 times smaller than B-SHARP on average. In both datasets, the noun and discourse counts are significantly different between the control and the other groups.

It is interesting that a significant difference is found in verbs whereas it is not the case for complex structures in the DementiaBank, which is opposite in B-SHARP. This may imply that the verb usage deteriorates as it progresses from MCI to dementia, but more thorough research is needed for further verification.

# An Exploratory Study on Multilingual Quality Estimation

Shuo Sun,<sup>1\*</sup> Marina Fomicheva,<sup>2\*</sup> Frédéric Blain,<sup>2</sup>

Vishrav Chaudhary,<sup>3</sup> Ahmed El-Kishky,<sup>3</sup> Adithya Renduchintala,<sup>3</sup> Francisco Guzmán,<sup>3</sup> Lucia Specia<sup>2,4</sup>

<sup>1</sup>Johns Hopkins University, <sup>2</sup>University of Sheffield, <sup>3</sup>Facebook AI

<sup>4</sup>Imperial College London

<sup>1</sup>ssun32@jhu.edu

<sup>2</sup>{m.fomicheva, f.blain, l.specia}@sheffield.ac.uk

<sup>3</sup>{vishrav, ahelk, adirendu, fguzman}@fb.com

## Abstract

Predicting the quality of machine translation has traditionally been addressed with language-specific models, under the assumption that the quality label distribution or linguistic features exhibit traits that are not shared across languages. An obvious disadvantage of this approach is the need for labelled data for each given language pair. We challenge this assumption by exploring different approaches to multilingual Quality Estimation (QE), including using scores from translation models. We show that these outperform single-language models, particularly in less balanced quality label distributions and low-resource settings. In the extreme case of zero-shot QE, we show that it is possible to accurately predict quality for any given new language from models trained on other languages. Our findings indicate that state-of-the-art neural QE models based on powerful pre-trained representations generalise well across languages, making them more applicable in real-world settings.

## 1 Introduction

Quality Estimation (QE) (Blatz et al., 2004a; Specia et al., 2009) is the task of predicting the quality of an automatically generated translation at test time, when no reference translation is available for comparison. Instead of reference translations, QE turns to explicit quality indicators that are either provided by the Machine Translation (MT) system itself (the so-called *glass-box* features) or extracted from both the source and the target texts (the so-called *black-box* features) (Specia et al., 2018b).

In the current QE approaches, black-box features are learned representations extracted by fine-tuning pre-trained multilingual or cross-lingual sentence encoders such as BERT (Devlin et al., 2018),

XLM-R (Conneau et al., 2019) or LASER (Artetxe and Schwenk, 2019). These supervised approaches have led to the state-of-the-art (SOTA) results in this task (Kepler et al., 2019; Fonseca et al., 2019), similarly to what has been observed for a myriad of other downstream natural language processing applications that rely on cross-lingual sentence similarity. Glass-box features are usually obtained by extracting various types of information from the MT system, e.g. lexical probability or language model probability in the case of statistical MT systems (Blatz et al., 2004b), or more recently softmax probability and attention weights from neural MT models (Fomicheva et al., 2020). Glass-box approach is potentially useful for low resource or zero-shot scenarios as it does not require large amounts of labelled data for training, but it does not perform as well as SOTA supervised models.

QE is therefore generally framed as a supervised machine learning problem, with models trained on data labelled for quality for each language pair. Training data publicly available to build QE models is constrained to very few languages, which has made it difficult to assess how well QE models generalise across languages. Therefore QE work to date has been addressed as a language-specific task.

The recent availability of multilingual QE data in a diverse set of language pairs (see Section 4.1) has made it possible to explore the multilingual potential of the QE task and SOTA models. In this paper, we posit that it is possible and beneficial to extend SOTA models to frame QE as a language-independent task.

We further explore the role of in-language supervision in comparison to supervision coming from other languages in a multi-task setting. Finally, we propose for the first time to model QE as a zero-shot cross-lingual transfer task, enabling new avenues of research in which multilingual models

\*Equal contribution.

can be trained once and then serve a multitude of languages.

The **main contributions** of this paper are: (i) we propose new multi-task learning approaches for multilingual QE (Section 3); (ii) we show that multilingual system outperforms single language ones (Section 5.1.1), especially in low-resource and less balanced label distribution settings (Section 5.1.3), and – counter-intuitively – that sharing a source or target language with the test case does not prove beneficial (Section 5.1.2); and (iii) we study black-box and glass-box QE in a multilingual setting and show that zero-shot QE is possible for both (Section 5.1.3 and 5.2).

## 2 Related Work

**QE** Early QE models were trained upon a set of explicit features expressing either the confidence of the MT system, the complexity of the source sentence, the fluency of the translation in the target language or its adequacy with regard to the source sentence (Specia et al., 2018b). Current SOTA models are learnt with the use of neural networks (NN) (Specia et al., 2018a; Fonseca et al., 2019). The assumption is that representations learned can, to some extent, account for source complexity, target fluency and source-target adequacy. These are fine-tuned from pre-trained word representations extracted using multilingual or cross-lingual sentence encoders such as BERT (Devlin et al., 2018), XLM-R (Conneau et al., 2019) or LASER (Artetxe and Schwenk, 2019).

Kim et al. (2017) propose the first breakthrough in neural-based QE with the *Predictor-Estimator* modular architecture. The *Predictor* model is an encoder-decoder Recurrent Neural Network (RNN) model trained on a huge amount of parallel data for a word prediction task. Its output is fed to the *Estimator*, a unidirectional RNN trained on QE data, to produce the quality estimates. Kepler et al. (2019) use a similar architecture where the Predictor model is replaced by pretrained contextualised word representations such as BERT (Devlin et al., 2018) or XLM-R (Conneau et al., 2019). Despite achieving strong performances, such models are resource heavy and need to be fine-tuned for each language-pair under consideration.

In a very different approach, Fomicheva et al. (2020) propose exploiting information provided by the NMT system itself. By exploring uncertainty quantification methods, they show that the

confidence with which the NMT system produces its translation correlates well with its quality. Although not performing as well as SOTA supervised models, their approach has the main advantage to be unsupervised and not rely on labelled data.

**Multilinguality** Multilinguality allows training a single model to perform a task from and to multiple languages. This principle has been successfully applied to NMT (Dong et al., 2015; Firat et al., 2016b,a; Nguyen and Chiang, 2017). Aharoni et al. (2019) stretches this approach by translating up to 102 languages from and to English using a Transformer model (Vaswani et al., 2017). They show that multilingual many-to-many models are effective in low resource settings. Multilinguality also allows for zero-shot translation (Johnson et al., 2017). With a simple encoder-decoder architecture and without explicit bridging between source and target languages, they show that their model is able to build a form of inter-lingual representation between all involved language pairs.

Shah and Specia (2016) is the only work in QE that attempted to explore models for more than one language. They use multitask learning with annotators or languages as multiple tasks. In a traditional black-box feature-based approach with Gaussian Processes as learning algorithm, their results suggest that adequately modelling the additional data is as important as the additional data itself. The multilingual models led to marginal improvements over bilingual ones. In addition, the experiments were only conducted with English translation into two closely related languages (French and Spanish).

## 3 Multilingual QE

In this section, we describe the QE models we propose and experiment with. They build upon pre-trained representations and represent the SOTA in QE, as we will show in Section 5.

**Pre-trained contextualised representations** such as BERT (Devlin et al., 2018) and XLM-R (Conneau et al., 2019) are deep contextualised language models based on the transformer neural architecture (Vaswani et al., 2017). These models are pre-trained on a large amount of texts in multiple languages and optimised with self-supervised loss functions. They use shared subword vocabularies that directly support more than a hundred languages without the need for

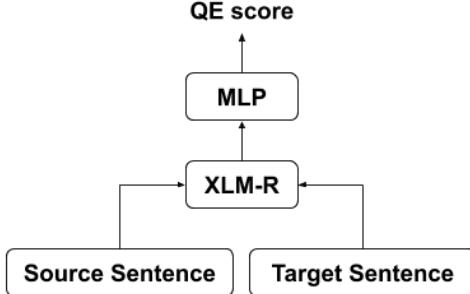


Figure 1: Baseline QE model.

any language-specific pre-processing. We explore QE models built on top of XLM-R, a pre-trained contextualised language model that achieves SOTA performance on multiple benchmark datasets.

**Baseline QE model (BASE)** Given a source sentence  $s^X$  in language  $X$  and a target sentence  $s^Y$  in language  $Y$ , we model the QE function  $f$  by stacking a 2-layer multilayer perceptron (MLP) on the vector representation of the [CLS] token from XLM-R:

$$\begin{aligned} f(s^X, s^Y) = & W_2 \cdot \text{ReLU}( \\ & W_1 \cdot E_{cls}(s^X, s^Y) + b_1 \quad (1) \\ & ) + b_2 \end{aligned}$$

where  $W_2 \in \mathbb{R}^{1 \times 4096}$ ,  $b_2 \in \mathbb{R}$ ,  $W_1 \in \mathbb{R}^{4096 \times 1024}$  and  $b_1 \in \mathbb{R}^{4096}$ .  $E_{cls}$  is a function that extracts the vector representation of the [CLS] token after encoding the concatenation of  $s^X$  and  $s^Y$  with XLM-R and ReLU is the Rectified Linear Unit activation function. We explore two training strategies: The **bilingual (BL)** strategy trains a QE model for every language pair while the **multilingual (ML)** strategy trains a single multilingual QE model for all language pairs, where the training data is simply pooled together without any language identifier. We note that this multilingual model here corresponds to a pooled, single-task learning approach.

**Multi-task Learning QE Model (MTL)** Multi-task learning has shown promising results in different NLP tasks (Ruder, 2017). Here, we want to explore whether having parameter sharing across languages is beneficial, and to what extent having language-specific predictors can boost performance. Therefore, we experiment with a simple multi-task approach where we concurrently optimise multiple QE BASE models that use a language-specific (LS) training strategy. To allow for testing in *zero-shot* conditions, we also train

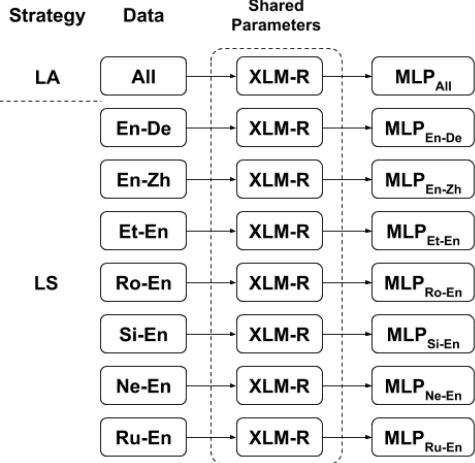


Figure 2: Multi-task learning QE model (MTL) with a shared XLM-R encoder.

a language-agnostic (LA) component, which receives sampled data from every language. We refer to these two models as **MTL-LA** and **MTL-LS**. As seen in Figure 2, the MTL-LS submodels and MTL-LA submodel share a common XLM-R encoder, while each submodel has its own dedicated language-specific MLP. The intuition of this approach is that it can result in improved learning efficiency and prediction accuracy by exploiting the similarities and differences in the QE tasks for different language directions (Thrun, 1996; Baxter, 2000). At training time, we iterate through the MTL-LS submodels in a round-robin fashion and alternate between training the MTL-LA submodel and training the chosen MTL-LS submodel. At test time, we can evaluate a test set with either the MTL-LA submodel or the MTL-LS submodel trained on the same language pair as the test set.

## 4 Experimental Setup

### 4.1 QE Dataset

We use the official data from the WMT 2020 QE Shared Task 1<sup>1</sup>. This dataset contains sentences extracted from Wikipedia (Fomicheva et al., 2020) and Reddit for Ru-En, translated to and from English for a total of 7 language pairs. The language pairs are divided into 3 categories: the high-resource English–German (En-De), English–Chinese (En-Zh) and Russian–English (Ru-En) pairs; the medium-resource Romanian–English (Ro-En) and Estonian–English (Et-En) pairs; and

<sup>1</sup><http://statmt.org/wmt20/quality-estimation-task.html>

the low-resource Sinhala–English (Si-En) and Nepali–English (Ne-En) pairs. Each translation was produced with SOTA transformer-based NMT models and manually annotated for quality using an annotation scheme inspired by the Direct Assessment (DA) methodology proposed by Graham et al. (2013). Specifically, translations were annotated on a 0-100 scale, where the 0-10 range represents an incorrect translation; 11-29, a translation with few correct keywords, but the overall meaning is different from the source; 30-50, a translation with major mistakes; 51-69, a translation which is understandable and conveys the overall meaning of the source but contains typos or grammatical errors; 70-90, a translation that closely preserves the semantics of the source sentence; and 90-100, a perfect translation. Figure 3 shows the distribution of DA scores for the different language pairs.

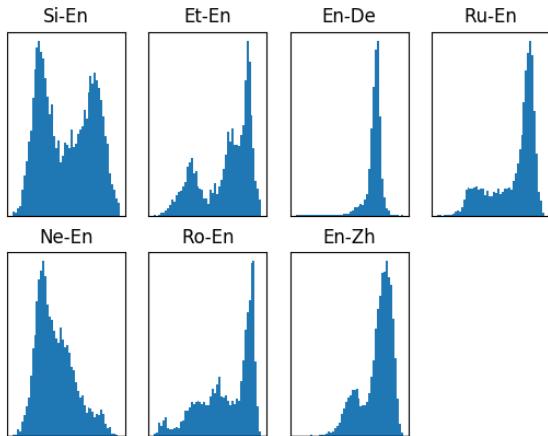


Figure 3: Distribution of DA judgments for different language pairs.

## 4.2 Settings

We train and test our models in the following conditions:

**Data splits** we use the training and development sets provided for the WMT2020 shared task on QE.<sup>2</sup> Since the test set is not publicly available, we further split the 7,000-instance training set for each language pair by using the first 6,000 instances for training and the last 1,000 instances for development, and report results on the official (1,000) development set.

**Training details** We optimise our models with Adam (Kingma and Ba, 2015) and use the same

<sup>2</sup><http://www.statmt.org/wmt20/quality-estimation-task.html>

learning rate ( $1e^{-6}$ ) for all experiments. We use a batch size of 8 and train on Nvidia V100 GPUs for 20 epochs. Each model is trained 5 times with different random seeds.

**Evaluation** All results in this paper are in terms of the average Pearson’s correlation for predicted QE scores against gold QE scores over the 5 different runs. Pearson correlation is the standard metric for this task, but we also compute error using Root Mean Squared Error (RMSE) (see Appendix).

## 5 Results

In what follows, we pose and discuss various hypotheses on multilinguality for QE. First we focus on our black-box approach from Section 3 (Section 5.1). Second, we examine the behavior of a glass-box approach which does not directly model the source and target texts in multilingual settings (Section 5.2). In all cases, we define  $\text{TrainL}$  as the set of language pairs used for training the QE model, and  $\text{TestL}$  as the set of language pairs used at test time.

### 5.1 Black-box QE Approach

#### 5.1.1 Multilingual models are better than bilingual models

As we can see from the results in Table 1<sup>3</sup>, the average Pearson’s correlation scores of the multilingual models are always higher than the bilingual ones, in some cases by a large margin. This is particularly true for En-De where the best BL model performs at Pearson’s correlation of 0.39, while both BASE-ML and MTL-LA achieve 0.47, which is a 20.5% relative improvement over the best BL model. Furthermore, the average score of Base-ML across all  $\text{TestL}$  is 0.69, 0.03 (4.5%) higher than the average score (0.66) of the best BASE-BL scores across all  $\text{TestL}$  (diagonal in the top part of Table 1). The results clearly show that multilingual models generally outperform bilingual models, even when the latter are optimised individually for different  $\text{TestL}$ . An interesting observation in Table 1 is that some BASE-BL models trained on different  $\text{TrainL}$  than  $\text{TestL}$  can perform almost as well as the models trained on the same  $\text{TrainL}$  as  $\text{TestL}$ . For example,

<sup>3</sup>The best results for BASE-BL are underlined and bold marks the best results across all models. Significant improvements over BASE-BL are marked with \*. We use the Hotelling-Williams test for dependent correlations to compute significance of the difference between correlations (Williams, 1959) with p-value < 0.05.

Model	Strategy	TrainL	TestL							
			En-De	En-Zh	Et-En	Ro-En	Si-En	Ne-En	Ru-En	Avg
BASE	BL	En-De	0.39	(-0.17)	(-0.39)	(-0.51)	(-0.32)	(-0.51)	(-0.35)	0.34
		En-Zh	(-0.02)	0.47	(-0.19)	(-0.36)	(-0.16)	(-0.24)	(-0.17)	0.50
		Et-En	(-0.10)	(-0.08)	<b>0.75</b>	(-0.20)	(-0.07)	(-0.10)	(-0.08)	0.57
		Ro-En	(-0.10)	(-0.14)	(-0.02)	<b>0.89</b>	(-0.02)	(-0.04)	(-0.08)	0.60
		Si-En	(-0.13)	(-0.13)	(-0.08)	(-0.15)	<b>0.66</b>	(-0.05)	(-0.07)	0.57
		Ne-En	(-0.10)	(-0.11)	(-0.06)	(-0.08)	(-0.01)	<b>0.77</b>	(-0.08)	0.60
		Ru-En	(-0.04)	(-0.09)	(-0.19)	(-0.26)	(-0.11)	(-0.16)	<b>0.70</b>	0.54
		ML	All	<b>0.47*</b>	<b>0.49</b>	<b>0.78*</b>	0.89	<b>0.70*</b>	0.78	<b>0.73</b>
MTL	LS	All	0.45	0.48	0.77	0.89	0.66	<b>0.79</b>	0.72	0.68
	LA	All	<b>0.47*</b>	<b>0.49</b>	0.76	0.89	0.66	0.78	0.72	0.68
	LS	En-*	0.41	0.46	-	-	-	-	-	-
	LA	En-*	0.45	0.46	-	-	-	-	-	-
	LS	*-En	-	-	<b>0.78*</b>	<b>0.90</b>	0.69	<b>0.79</b>	<b>0.73</b>	-
	LA	*-En	-	-	<b>0.78*</b>	0.89	0.69	0.78	<b>0.73</b>	-
‡ BERT-BiRNN (Fomicheva et al., 2020)		0.27	0.37	0.64	0.76	0.47	0.55	-	-	-
‡ WMT20 QE Shared Task 1 Leaderboard (June 2020)		0.47	0.48	0.79	0.90	0.65	0.79	0.78	0.69	-

Table 1: Results for BASE and MTL QE models. We train different BASE-BL models for every language pair and a single BASE-ML model on all language pairs. We also train a single MTL QE model consists of multiple MTL-LS and MTL-LA submodels. For each TestL, we evaluate it with the MTL-LS submodel trained on the same language pair. We bold the best results across all models. Significant improvements over BASE BL are marked with \*. ‡ identifies systems trained on the full 7,000-instance training set with performances reported on the official test set of the WMT’20 QE Shared Task 1 (<https://competitions.codalab.org/competitions/24447>), which we assume to come from the same distribution as the dev set.

a BASE-BL model trained on En-Zh and tested on En-De performs at average Pearson’s correlation of 0.37, which is only 0.02 below the best result. We hypothesize that XLM-R might be capturing certain traits in TrainL that can generalise well to other TestL, i.e. the complexity of source sentences or the fluency of the target sentences (Sun et al., 2020).

### 5.1.2 There is little benefit from specialisation

Here we investigate whether having specialised language-specific sub-models which can benefit from the shared supervision from other languages while keeping their focus on a language-specific task can help to improve performance. Furthermore, it is possible that multi-task learning works better when language pairs share certain characteristics. Therefore, we also investigate whether combining language pairs that share either source or target languages can be more beneficial. For that, we use the MTL models but with a reduced set of languages.

From the results in Table 1, we observe that language-specialised predictors do not help improve performance. There is no clear advantage in using the multi-task learning QE approach (MTL-

LS and MTL-LA) where each language pair is treated as a separate task; over the simple single-task multi-lingual learning approach (BASE-ML), despite the former having more parameters and language-specific MLP layers.

In the table, we compare MTL models trained on language pairs that share the source language (En-\*) or the target language (\*-En) against MTL models trained on all languages (All). As we can see from the results, the MTL model trained on En-\* perform worse than the MTL model trained on all language pairs. In contrast, the MTL model trained on \*-En performs a little bit better than the MTL model trained on all language pairs on 4 out of the 5 language pairs and is comparable to Base-ML on those language directions.

### 5.1.3 Multilingual models help zero- and few-shot QE

To test whether a multilingual model for QE can generalise beyond the language pairs observed during training, we also conduct experiments varying amounts of in-language data (i.e. 0% –zero-shot, 5%, 10%, 25%, 50%, 75% and 100%). We build and compare BASE-BL and BASE-ML models. We train BASE-BL models only on the sub-

% in-lang	Model	Strategy	TestL							
			En-De	En-Zh	Et-En	Ro-En	Si-En	Ne-En	Ru-En	Avg
<u>0</u>	<b>BASE</b>	ML	0.45	0.42	0.75	0.80	0.68	0.76	0.68	<u>0.65</u>
		BL	0.13	0.39	0.65	0.70	0.58	0.63	0.63	0.53
<u>5</u>	<b>BASE</b>	ML	<u>0.38</u>	<u>0.44</u>	<u>0.74</u>	<u>0.85</u>	<u>0.67</u>	<u>0.76</u>	<u>0.71</u>	<u>0.65</u>
		BL	0.24	0.43	0.69	0.85	0.56	0.68	0.64	0.58
<u>10</u>	<b>BASE</b>	ML	<u>0.37</u>	<u>0.46</u>	<u>0.75</u>	<u>0.87</u>	<u>0.64</u>	<u>0.77</u>	<u>0.71</u>	<u>0.65</u>
		BL	0.27	0.45	0.70	0.87	0.61	0.72	0.70	0.62
<u>25</u>	<b>BASE</b>	ML	<u>0.40</u>	<u>0.46</u>	<u>0.75</u>	<u>0.88</u>	<u>0.66</u>	<u>0.76</u>	<u>0.71</u>	<u>0.66</u>
		BL	0.33	0.47	0.74	0.88	0.62	0.74	0.69	0.64
<u>50</u>	<b>BASE</b>	ML	<u>0.41</u>	<u>0.48</u>	<u>0.76</u>	<u>0.89</u>	<u>0.69</u>	<u>0.77</u>	<u>0.72</u>	<u>0.67</u>
		BL	0.39	0.47	0.75	0.88	0.64	0.76	0.70	0.66
<u>75</u>	<b>BASE</b>	ML	<u>0.46</u>	<u>0.49</u>	<u>0.78</u>	<u>0.89</u>	<u>0.70</u>	<u>0.78</u>	<u>0.71</u>	<u>0.69</u>
		BL	0.39	0.47	0.75	0.89	0.66	0.77	0.70	0.66
<u>100</u>	<b>BASE</b>	ML	<u>0.47</u>	<u>0.49</u>	<u>0.78</u>	<u>0.89</u>	<u>0.70</u>	<u>0.78</u>	<u>0.73</u>	<u>0.69</u>

Table 2: Results of BASE QE models for different portions of training data (%data). For BASE-ML, we train the models on subsampled training data in the test language pair and all training data in other language pairs. For BASE-BL, we train the models on only subsampled training data in the test language pair. We underline the best results for each %data setting.

sampled in-language training data and train BASE-ML on both sub-sampled in-language training data and all training data in other language pairs. In other words, we want to know whether multilingual QE helps if we have limited or no training data in our desired test language pair. Results are shown in Table 2. For ease of visualisation, we also plot the Pearson’s correlation results against the percentage of in-language training data in Figure 4. As seen in Table 2, the multilingual model performs better than the bilingual models on all language pairs for every configuration of training data. Moreover, in 3 out of 7 cases, the *zero-shot* models perform better than the fully-trained bilingual models. This provides strong evidence that the QE task can be solved in a multilingual way, without loss of performance compared to bilingual performance. It also shows strong evidence for the zero-shot applicability of our models.

## 5.2 Glass-box QE Approach

Having pre-trained representations can help build state-of-the-art multilingual systems. However, these representations are costly to compute in practice, which limits their applicability for building QE systems for real-time scenarios. Glass-box approaches to QE extract information from the NMT system itself to predict quality, without directly relying on the source and target text or using any external resources. To test how well this information can generalise across different languages, we lever-

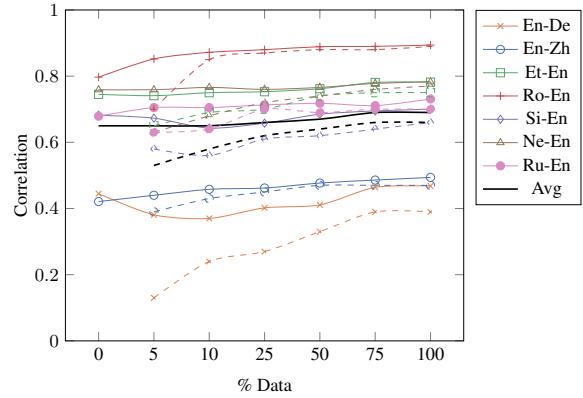


Figure 4: Results of BASE QE models for various zero-shot and few-shot cross-lingual transfer settings. The solid lines represent the BASE ML models while the dashed lines are the BASE BL models.

age existing work on glass-box QE by Fomicheva et al. (2020) that explores NMT output distribution to capture predictive uncertainty as a proxy for MT quality. We use the following 5 best-performing glass-box indicators from their work:

- Average NMT log-probability of the translated sentence;
- Variance of word-level log-probabilities;
- Entropy of NMT softmax output distribution;
- NMT log-probability of translations generated with Monte Carlo dropout (Gal and Ghahramani, 2016);<sup>4</sup>

<sup>4</sup>This method consists in performing several forward

TrainL	TestL					
	En-De	En-Zh	Et-En	Ro-En	Si-En	Ne-En
<b>En-De</b>	0.24	(-0.25)	(-0.36)	(-0.22)	(-0.24)	(-0.32)
<b>En-Zh</b>	(+0.08)	0.44	(-0.05)	(-0.04)	(-0.03)	(-0.08)
<b>Et-En</b>	(+0.07)	(-0.03)	<b>0.61</b>	(-0.02)	(-0.02)	(-0.06)
<b>Ro-En</b>	(+0.05)	(-0.05)	(-0.03)	<b>0.76</b>	(-0.02)	(-0.06)
<b>Si-En</b>	(+0.06)	(-0.04)	(-0.04)	(-0.03)	<b>0.54</b>	(-0.03)
<b>Ne-En</b>	(-0.00)	(-0.09)	(-0.09)	(-0.09)	(-0.04)	<b>0.58</b>
<b>All langs</b>	0.32	0.44	0.60	0.75	0.55	0.56
<b>Best feature</b>	0.26	0.32	0.64	0.69	0.51	0.60

Table 3: Pearson correlation for regression models based on glass-box features trained on each language pair and evaluated either on the same language pair or other language pairs. For testing on a different language pair we report the difference in Pearson correlation with respect to training and testing on the same language pair. For comparison we show the correlation individual best performing feature with no learning involved.

- Lexical similarity between MT hypotheses generated with Monte Carlo dropout.

We train an XGboost regression model (Chen and Guestrin, 2016)<sup>5</sup> to combine these features to predict DA judgments and test the performance of the model in multilingual settings. Table 3 shows Pearson correlation for the regression models trained on each language pair and evaluated either on the same language pair or other language pairs.<sup>6</sup> The ‘All langs’ row indicates the results when training on all language pairs, whereas ‘Best feature’ indicates the correlation obtained by the best performing feature individually. Comparing these results to the results for pre-trained representations in Table 1 we can make three observations.

### 5.2.1 Glass-box features are more comparable across languages

First, although the correlation is generally lower for the glass-box approach, performance degradation when testing on different language pairs is smaller. For all language pairs except English-German, we observe a relatively small decrease in performance (up to 0.09) when training and test language pairs are different. This suggests that the indicators extracted from the NMT model are more

passes through the network, collecting posterior probabilities generated by the model with parameters perturbed by dropout and using the resulting distribution to approximate model uncertainty.

<sup>5</sup>We chose a regression model over an NN given the smaller number of features available.

<sup>6</sup>These experiments do not include Russian-English, as the corresponding NMT system is an ensemble and it is not evident how the glass-box features proposed by Fomicheva et al. (2020) should be extracted in this case.

comparable across languages than input features from pre-trained representations.

We note that the NMT systems in MLQE dataset were all based on Transformer architecture but trained using different amount of data and have different overall output quality. Interestingly, the results of this experiment indicate that glass-box information extracted from these systems could be language-independent. More experiments are needed to confirm if this observation can be extrapolated to other datasets, language pairs, domains and MT systems.

### 5.2.2 Multilingual gains are limited by learning algorithm

Second, by contrast to the results in Table 1 where multilingual training brings significant improvements, we do not see any gains in performance from training with all available data. The reason could be that training a regression model with a small number of features does not require large amounts of training data, and therefore performance does not improve with additional data. English-German is an exception with a large gain in correlation when training on all language pairs.

### 5.2.3 The output label distribution matters

Finally, similarly to the black-box approach in Table 1, the performance for English-German benefits from using the data from other language pairs for training. This indicates that the results are affected by factors that are independent of the approach used for prediction. To better understand these results we look at the distribution of NMT log-probabilities (Figure 5) and the distribution of DA scores (Figure 3).

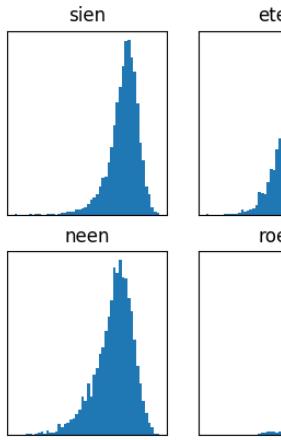


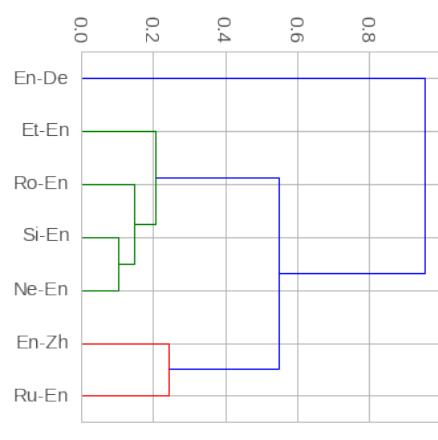
Figure 5: Distribution of NMT log-probabilities for different language pairs

While log-probability distributions are comparable across language pairs, the distributions of DA scores are very different. We suggest, therefore, that the decrease in performance when testing on a different language is related to a higher extent to the shift in the output distribution across languages (i.e. DA judgments) than to the shift in the input features. This also explains the difficulty for training and predicting on English-German data where the distribution of DA scores is highly skewed with minimal variability in the quality range.

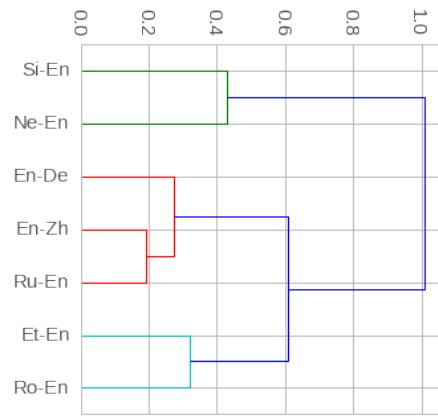
## 6 Discussion and Conclusions

From our various experiments, one setting that stood out is that of English-German. We suggest that the difficulty for predicting quality for this language pair was exacerbated by the metric used for evaluation. Because of its sample-dependence, Pearson correlation can be more sensitive to the output distribution. In contrast, an error-based metric like RMSE will be less sensitive to these variations. To illustrate these effects, in Figure 6, we show the hierarchical clustering of language directions obtained by using the metric value from training on one direction and testing on another one as a notion of distance. In subfigure (a), we observe the clusters based on Pearson correlation as shown in Table 1. In subfigure (b), we observe the same clustering done based on RMSE. It should be noted that in the former, En-De is a clear outlier, whereas in the latter, we have a clustering that is more consistent with the general maturity of the language pairs: Ne-En and Si-En are *low resource*, Ro-En and Et-En are *medium resource*, etc.

We explored the use of multilingual contextual



(a) Pearson correlation



(b) RMSE

Figure 6: Language hierarchical clustering according to the results of training on one language and testing on another. In subfigure (a) we plot the clustering based on Pearson correlation. In subfigure (b) we plot the same clustering based on RMSE. The y axis denotes the *distance* between language pairs according to each evaluation.

representations to build state-of-the-art multilingual QE models. From our experiments, we observed that: 1) multilingual systems are *always* better than bilingual systems; 2) having multi-task models, which share parts of the model across languages and specialise others, does not necessarily yield better results; and 3) multilingual systems for QE generalise well across languages and are powerful even in *zero-shot* scenarios. We also contrasted the use of pre-trained representations which are costly to obtain, to the use of glass-box features which can be extracted from the NMT system. We observed that glass-box features are very comparable across languages, and training multilingual systems with them adds little value. Finally, we observed that the distribution of the output labels matters for the evaluation of QE.

## Acknowledgments

Marina Fomicheva, Frédéric Blain and Lucia Specia were supported by funding from the Bergamot project (EU H2020 Grant No. 825303).

## References

- Roe Aharoni, Melvin Johnson, and Orhan Firat. 2019. [Massively multilingual neural machine translation](#). *arXiv preprint arXiv:1903.00089*.
- Mikel Artetxe and Holger Schwenk. 2019. [Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond](#). *Transactions of the Association for Computational Linguistics*, 7:597–610.
- Jonathan Baxter. 2000. [A model of inductive bias learning](#). *Journal of artificial intelligence research*, 12:149–198.
- John Blatz, Erin Fitzgerald, George Foster, Simona Gandrabur, Cyril Goutte, Alex Kulesza, Alberto San-chis, and Nicola Ueffing. 2004a. [Confidence estimation for machine translation](#). In *Proceedings of the 20th International Conference on Computational Linguistics*, Geneva, Switzerland.
- John Blatz, Erin Fitzgerald, George Foster, Simona Gandrabur, Cyril Goutte, Alex Kulesza, Alberto San-chis, and Nicola Ueffing. 2004b. Confidence estimation for machine translation. In *COLING*.
- Tianqi Chen and Carlos Guestrin. 2016. [Xgboost: A scalable tree boosting system](#). In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD ’16, page 785–794, New York, NY, USA. Association for Computing Machinery.
- Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. [Unsupervised cross-lingual representation learning at scale](#). *arXiv preprint arXiv:1911.02116*.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. [Bert: Pre-training of deep bidirectional transformers for language understanding](#). *arXiv preprint arXiv:1810.04805*.
- Daxiang Dong, Hua Wu, Wei He, Dianhai Yu, and Haifeng Wang. 2015. [Multi-task learning for multiple language translation](#). In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1723–1732, Beijing, China. Association for Computational Linguistics.
- Orhan Firat, Kyunghyun Cho, and Yoshua Bengio. 2016a. [Multi-way, multilingual neural machine translation with a shared attention mechanism](#). *ArXiv*, abs/1601.01073.
- Orhan Firat, Baskaran Sankaran, Yaser Al-Onaizan, Fatos T. Yarman-Vural, and Kyunghyun Cho. 2016b. [Zero-resource translation with multi-lingual neural machine translation](#). *ArXiv*, abs/1606.04164.
- Marina Fomicheva, Shuo Sun, Lisa Yankovskaya, Frédéric Blain, Francisco Guzmán, Mark Fishel, Nikolaos Aletras, Vishrav Chaudhary, and Lucia Specia. 2020. [Unsupervised quality estimation for neural machine translation](#). *arXiv preprint arXiv:2005.10608*.
- Erick Fonseca, Lisa Yankovskaya, André FT Martins, Mark Fishel, and Christian Federmann. 2019. [Findings of the wmt 2019 shared tasks on quality estimation](#). In *Proceedings of the Fourth Conference on Machine Translation (Volume 3: Shared Task Papers, Day 2)*, pages 1–10.
- Yarin Gal and Zoubin Ghahramani. 2016. [Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning](#). In *International Conference on Machine Learning*, pages 1050–1059.
- Yvette Graham, Timothy Baldwin, Alistair Moffat, and Justin Zobel. 2013. [Continuous measurement scales in human evaluation of machine translation](#). In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 33–41.
- Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda B. Viégas, Martin Wattenberg, Gregory S. Corrado, Macduff Hughes, and Jeffrey Dean. 2017. [Google’s multilingual neural machine translation system: Enabling zero-shot translation](#). *Transactions of the Association for Computational Linguistics*, 5:339–351.
- Fábio Kepler, Jonay Trénous, Marcos Treviso, Miguel Vera, António Góis, M Amin Farajian, António V Lopes, and André FT Martins. 2019. [Unbabel’s participation in the wmt19 translation quality estimation shared task](#). *WMT 2019*, page 80.
- Hyun Kim, Jong-Hyeok Lee, and Seung-Hoon Na. 2017. [Predictor-estimator using multilevel task learning with stack propagation for neural quality estimation](#). In *Proceedings of the Second Conference on Machine Translation*, pages 562–568.
- Diederik P. Kingma and Jimmy Ba. 2015. [Adam: A method for stochastic optimization](#). In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.
- Toan Q. Nguyen and David Chiang. 2017. [Transfer learning across low-resource, related languages for neural machine translation](#). *ArXiv*, abs/1708.09803.

Sebastian Ruder. 2017. [An overview of multi-task learning in deep neural networks](#). *arXiv preprint arXiv:1706.05098*.

Kashif Shah and Lucia Specia. 2016. [Large-scale multitask learning for machine translation quality estimation](#). In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 558–567.

Lucia Specia, Frédéric Blain, Varvara Logacheva, Ramón Astudillo, and André F. T. Martins. 2018a. [Findings of the wmt 2018 shared task on quality estimation](#). In *Proceedings of the Third Conference on Machine Translation, Volume 2: Shared Task Papers*, pages 702–722, Belgium, Brussels. Association for Computational Linguistics.

Lucia Specia, Nicola Cancedda, Marc Dymetman, Marco Turchi, and Nello Cristianini. 2009. [Estimating the sentence-level quality of machine translation systems](#). In *Proceedings of the 13th Annual Conference of the European Association for Machine Translation*, pages 28–35, Barcelona, Spain.

Lucia Specia, Carolina Scarton, and Gustavo Henrique Paetzold. 2018b. [Quality Estimation for Machine Translation](#). Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Shuo Sun, Francisco Guzmán, and Lucia Specia. 2020. [Are we estimating or guesstimating translation quality?](#) In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6262–6267, Online. Association for Computational Linguistics.

Sebastian Thrun. 1996. [Is learning the n-th thing any easier than learning the first?](#) In *Advances in neural information processing systems*, pages 640–646.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. [Attention is all you need](#). In *Advances in neural information processing systems*, pages 5998–6008.

Evan James Williams. 1959. *Regression Analysis*, volume 14. Wiley, New York, USA.

## Appendix

For completeness, Tables 4 and 5 report RMSE scores for our main experiments.

Model	Strategy	TrainL	TestL							
			En-De	En-Zh	Et-En	Ro-En	Si-En	Ne-En	Ru-En	Avg
BASE	BL	En-De	<u>0.71</u>	0.72	0.86	0.92	0.79	0.82	0.88	0.81
		En-Zh	0.85	<u>0.69</u>	0.75	0.85	0.75	0.83	0.82	0.79
		Et-En	0.76	<u>0.69</u>	<u>0.59</u>	0.71	0.78	0.91	0.74	0.74
		Ro-En	0.93	0.77	0.61	<u>0.48</u>	0.82	1.01	0.79	0.77
		Si-En	1.01	0.79	0.89	0.94	<u>0.64</u>	0.61	0.87	0.82
		Ne-En	1.13	0.84	1.10	1.16	0.79	<u>0.57</u>	1.05	0.95
		Ru-En	0.83	0.66	0.78	0.87	0.73	0.73	<u>0.67</u>	0.75
		ML	All	<b>0.68</b>	0.65	<b>0.55</b>	<b>0.44</b>	<b>0.59</b>	<b>0.53</b>	<b>0.65</b>
MTL	LS	All	0.69	<b>0.64</b>	0.56	0.45	0.62	0.54	0.66	0.59
	LA	All	<b>0.68</b>	<b>0.64</b>	0.57	<b>0.44</b>	0.61	0.54	0.66	0.59
	LS	En-*	0.71	0.70	-	-	-	-	-	-
	LA	En-*	0.69	0.68	-	-	-	-	-	-
	LS	*-En	-	-	0.56	0.46	0.60	0.55	0.64	-
	LA	*-En	-	-	0.56	0.46	0.61	0.54	0.66	-

Table 4: RMSE for BASE and MTL QE models. We underline the best RMSE for BASE-BL and bold the best RMSE across all models.

TestL										
%data	Model	Strategy	En-De	En-Zh	Et-En	Ro-En	Si-En	Ne-En	Ru-En	Avg
0	BASE	ML	0.74	0.65	0.64	0.65	0.61	0.84	0.72	0.69
		BL	<u>0.74</u>	<u>0.70</u>	0.72	0.75	0.76	0.76	0.74	0.74
5	BASE	ML	0.77	0.74	<u>0.62</u>	<u>0.56</u>	<u>0.73</u>	<u>0.62</u>	<u>0.71</u>	<u>0.68</u>
		BL	<u>0.74</u>	<u>0.70</u>	0.71	0.59	0.74	0.71	0.75	0.71
10	BASE	ML	0.77	0.72	<u>0.62</u>	<u>0.54</u>	<u>0.73</u>	<u>0.64</u>	<u>0.70</u>	<u>0.67</u>
		BL	<u>0.74</u>	<u>0.70</u>	0.71	0.59	0.74	0.71	0.75	0.71
25	BASE	ML	0.72	0.71	<u>0.61</u>	<u>0.49</u>	<u>0.69</u>	<u>0.64</u>	<u>0.70</u>	<u>0.65</u>
		BL	0.77	<u>0.70</u>	0.65	0.54	0.74	0.70	0.71	0.69
50	BASE	ML	<u>0.69</u>	<u>0.68</u>	<u>0.59</u>	<u>0.47</u>	<u>0.65</u>	<u>0.59</u>	<u>0.67</u>	<u>0.62</u>
		BL	0.73	0.72	0.60	0.52	0.68	0.62	0.71	0.65
75	BASE	ML	<u>0.67</u>	<u>0.65</u>	<u>0.55</u>	<u>0.45</u>	<u>0.62</u>	<u>0.54</u>	<u>0.67</u>	<u>0.59</u>
		BL	0.71	0.70	0.59	0.48	0.65	0.61	0.68	0.63
100	BASE	ML	<u>0.68</u>	<u>0.66</u>	<u>0.56</u>	<u>0.44</u>	<u>0.60</u>	<u>0.54</u>	<u>0.65</u>	<u>0.59</u>
		BL	0.72	0.68	0.57	0.47	0.64	0.56	0.68	0.62

Table 5: RMSE of BASE QE models for different portions of training data (%data). We underline the best RMSE for each %data setting.

# English-to-Chinese Transliteration with a Phonetic Auxiliary Task

Yuan He\*

Department of Computer Science  
University of Oxford  
yuan.he@cs.ox.ac.uk

Shay B. Cohen

School of Informatics  
University of Edinburgh  
scohen@inf.ed.ac.uk

## Abstract

Approaching named entities transliteration as a Neural Machine Translation (NMT) problem is common practice. While many have applied various NMT techniques to enhance machine transliteration models, few focus on the linguistic features particular to the relevant languages. In this paper, we investigate the effect of incorporating phonetic features for English-to-Chinese transliteration under the multi-task learning (MTL) setting—where we define a phonetic auxiliary task aimed to improve the generalization performance of the main transliteration task. In addition to our system, we also release a new English-to-Chinese dataset and propose a novel evaluation metric which considers multiple possible transliterations given a source name. Our results show that the multi-task model achieves similar performance as the previous state of the art with a model of a much smaller size.<sup>1</sup>

## 1 Introduction

Transliteration, the act of mapping a name from the orthographic system of one language to another, is directed by the pronunciation in the source and target languages, and often by historical reasons or conventions. It plays an important role in tasks like information retrieval and machine translation (Marton and Zitouni, 2014; Hermjakob et al., 2008).

Over the recent years, many have addressed transliteration using sequence-to-sequence (seq2seq) deep learning models (Rosca and Breuel, 2016; Merhav and Ash, 2018; Grundkiewicz and Heafield, 2018), enhanced with several NMT techniques (Grundkiewicz and Heafield, 2018). However, this recent work neglects the most crucial feature for transliteration, i.e. pronunciation. To

English	IPA	Chinese	Pinyin
A	/eɪ/	艾	ài
my	/mi/	米	mǐ

Table 1: An example of English-to-Chinese transliteration, from *Amy* to 艾米. Each row presents a group of corresponding subsequences in different representations.

bridge this gap, we define a phonetic auxiliary task that shares the sound information with the main transliteration task under the multi-task learning (MTL) setting.

Depending on the specific language, the written form of a word reveals its pronunciation to various extents. For alphabetical languages such as English and French, a letter, or a sequence of letters, usually reflects the word pronunciation. For example, the word *Amy* (in the International Phonetic Alphabet, IPA, /'eɪ.mi/) has the sub-word *A* corresponding to /'eɪ/ and *my* corresponding to /mi/. In contrast, characters in a logographic<sup>2</sup> writing system for languages like Chinese or Japanese do not explicitly indicate sound (Xing et al., 2006).

In this paper, we give a treatment to the problem of transliteration from English (alphabet) to Chinese<sup>3</sup> (logogram) using an RNN-based MTL model with a phonetic auxiliary task. We transform each Chinese character to the alphabetical representation of its pronunciation via the official phonetic writing system, Pinyin,<sup>4</sup> which uses Latin letters with four diacritics denoting tones to represent the sounds.

<sup>2</sup>A logogram is an individual character that represents a whole word or phrase.

<sup>3</sup>The Chinese language we mention in this paper refers explicitly to Mandarin, which is the official language originated from the northern dialect in China.

<sup>4</sup>Pinyin is the official romanization system for Standard Chinese (Mandarin) in mainland China and to some extent in Taiwan. It does not apply to other Chinese dialects.

\*Work done at The University of Edinburgh.

<sup>1</sup>Our code and data are available at <https://github.com/Lawhy/Multi-task-NMTransliteration>.

For example, the Chinese transliteration for *Amy* is 艾米 and the associated Pinyin representation is *ài mǐ*. We summarize the correspondences occurring in this example in Table 1.

Due to the similarity between the source name and the Pinyin representation, Jiang et al. (2009) proposed a sequential transliteration model that uses Pinyin as an intermediate representation before transliterating a Chinese name to English. In contrast, our idea is to build a model with a shared encoder and dual decoders, that can learn the mapping from English to Chinese and Pinyin simultaneously. By jointly learning source-to-target and source-to-sound mappings, the encoder is expected to generalize better (Ruder, 2017) and pass more refined information to the decoders.

Transliteration datasets are often extracted from dictionaries, or aligned corpus generated from applying named entity recognition (NER) system to parallel newspaper articles in different languages (Sproat et al., 2006). We use two datasets for our experiments, one taken from NEWS Machine Transliteration Shared Task (Chen et al., 2018) and the other extracted from a large dictionary. We evaluate the transliteration system using both the conventional word accuracy and a novel metric designed for English-to-Chinese transliteration (see Section 5). Our contributions are as follows:

1. We make available a new English-to-Chinese named entities dataset (“DICT”) particular to names of people. This dataset is based on the dictionary *A Comprehensive Dictionary of Names in Roman-Chinese* (Xinhua News Agency, 2007).
2. We propose a substitution-based metric called Accuracy with Alternating Character Table (ACC-ACT), which gives a better estimation of the system’s quality than the traditional word accuracy (ACC).
3. We propose a multi-task learning transliteration model with a phonetic auxiliary task, and run experiments to demonstrate that it attains better scores than single-main-task or single-auxiliary-task models.

We report accuracy and F-score of 0.299 and 0.6799, respectively, on the NEWS dataset, with a model of size 22M parameters, compared to the previous state of the art (Grundkiewicz and Heafield, 2018), which achieves accuracy and F-score of 0.304 and 0.6791, respectively, with a model of size 133M parameters. On the DICT dataset, for

Source (x)	Target (y)	Pinyin (p)
Caleigh	凯莉	kai li

Table 2: An example data point under our multi-task learning setting.

the same model sizes, we report accuracy of 0.729 as compared to their 0.732.

## 2 Problem Formulation

We use the word *vocabulary* to describe the *set of characters* for the purpose of our task specification. Let  $V_{src}$  and  $V_{tgt}$  denote the source and target vocabularies, respectively. For a source word  $\mathbf{x}$  of length  $I$  and a target word  $\mathbf{y}$  of length  $J$ , we have:

$$\begin{aligned}\mathbf{x} &= (x_1, x_2, \dots, x_I) \in V_{src}^I, \\ \mathbf{y} &= (y_1, y_2, \dots, y_J) \in V_{tgt}^J.\end{aligned}$$

where the  $k$ th element in the vector denotes a character at position  $k$ .

We formulate the task of transliteration as a supervised learning problem: given a collection of  $n$  training examples,  $\{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})\}_{i=0}^n$ , the objective is to learn a predictor function,  $f : \mathbf{x} \rightarrow \mathbf{y}$ , of which the parameter space maximizes the following conditional probability:

$$P(\mathbf{y}|\mathbf{x}) \stackrel{\text{Chain Rule}}{=} \prod_{j=1}^J P(y_j|y_1, \dots, y_{j-1}, \mathbf{x}).$$

For our multi-task transliteration model, the predictor becomes  $f_{MTL} : \mathbf{x} \rightarrow (\mathbf{y}, \mathbf{p})$ , where  $\mathbf{p}$  denotes the written representation of the pronunciation of the target word  $\mathbf{y}$ . For decoding, we maximize the conditional probabilities,  $P(\mathbf{p}|\mathbf{x}, \tilde{\mathbf{y}})$  and  $P(\mathbf{y}|\mathbf{x}, \tilde{\mathbf{p}})$ , where  $\tilde{\mathbf{y}}$  and  $\tilde{\mathbf{p}}$  refers to the implicit information channeled by one task to the other.

The phonetic information we use for our task refers to the Pinyin version of the name in Chinese, without tone marks,<sup>5</sup> because they are often removed for spelling Chinese names in an alphabetical language. We present an example data point in the form of  $(\mathbf{x}, \mathbf{y}, \mathbf{p})$  in Table 2.

## 3 Dataset Preparation

We experiment with two different English-to-Chinese datasets. For simplicity, we denote the one

<sup>5</sup>For example, the Pinyins, *chī*, *chí*, *chǐ* and *chì*, are all transformed to *chi*. Note that this process will decrease the vocabulary size.

taken from NEWS Machine Transliteration Shared Task (Chen et al., 2018) as “NEWS,” and the one extracted from the dictionary (Xinhua News Agency, 2007) as “DICT.”

### 3.1 NEWS Dataset

We use the preprocessing script<sup>6</sup> created by Grundkiewicz and Heafield (2018) to construct the NEWS dataset from raw data provided in the Shared Task (Chen et al., 2018). This script merges the raw English-to-Chinese and Chinese-to-English datasets into a single one, then transforms it to uppercase<sup>7</sup> and tokenizes all names into sequences of characters (words are treated as sentences, characters are treated as words). In addition, it takes 513 examples from the training data to form the internal development set and uses the official development set as the internal test set.

To make the final comparison, we download the source-side data of the official test set from the Shared Task’s website,<sup>8</sup> and submit the transliteration results (see Section 6.4).

### 3.2 DICT Dataset

The source dictionary contains approximately 680K name pairs for transliteration from other languages than Chinese. We extracted 58,456 pairs that originated in English and performed the following preprocessing steps:

1. For the source side (English), we remove the inverted commas and white spaces from names that contain them (e.g. *A’Court, Le Gresley*).
2. For both sides, we lowercase<sup>9</sup> all the words and tokenize them into sequences of characters.
3. Name pairs with multiple target transliterations are removed from the dataset and saved in a separate file for the construction of the ACT (see next paragraph). As such, every name pair becomes unique in our preprocessed dataset. We randomly divide the rest into the ratio of 8 : 1 : 1, to form training, development and test sets.

We report the final partitions of both datasets in Table 3.

<sup>6</sup>Available at <https://github.com/snukky/news-translit-nmt>.

<sup>7</sup>We lowercase all the words in both NEWS and DICT datasets as evaluating transliteration is case-insensitive.

<sup>8</sup>The official test set with task ID T-EnCh is available at: <http://workshop.colips.org/news2018/dataset.html>.

<sup>9</sup>Lowercasing does not affect Chinese characters as they are not alphabetical.

Source	Train	Dev	Test
NEWS	81,252	513	1,000
DICT	46,620	5,828	5,828

Table 3: Numbers of data points in training, development and test sets of NEWS and DICT datasets. Dev and Test for the NEWS dataset (first row) refer to the internal development and test set, respectively.

### 3.3 Alternating Character Table

Chinese characters<sup>10</sup> that sound alike can often replace each other in the transliteration of a name from other languages. Unlike an alphabetical language where a similar pronunciation is bounded to sub-words of various lengths, characters in Chinese have concrete and independent pronunciations. Thus, we can conveniently build the Alternating Character Table (ACT) with each row storing a list of interchangeable characters.

We construct the ACT based on the DICT dataset because it contains less noise after applying significant data cleansing. In total, 449 English names from the DICT dataset have more than one transliterations in Chinese. We purposely removed all these names from the DICT data during the preprocessing so as to ensure that we are not using any knowledge from the test set. The final ACT contains 29 rows (see Appendix) and we use it with our adaptive evaluation metric (see Section 5).

### 3.4 Pinyin Conversion

In transliteration, the pronunciations of the Chinese characters are often unique (even for a polyphonic character, e.g. 什, that has more than one Pinyins, shí and shén, only shí is commonly used in transliteration). Therefore, we can directly transform each Chinese character into a unique Pinyin, thus forming the target data for the auxiliary task. The procedure is as follows: for each character  $y_t$  in the target name  $y$ , we use the Python package pypinyin<sup>11</sup> to map  $y_t$  to the corresponding Pinyin (without the tone mark). The tool will generate the most frequently used Pinyin for each  $y_t$  based on dictionary data. We then apply further manual correction on the Pinyins because the most frequent Pinyin is not necessarily the one used in transliteration.

<sup>10</sup>Limited to the set of characters (with size  $\approx 1K$  out of 80K) commonly used in transliteration.

<sup>11</sup>Available at: <https://github.com/mozillazg/python-pinyin>. We use the lazy\_pinyin feature to generate Pinyins without tone marks.

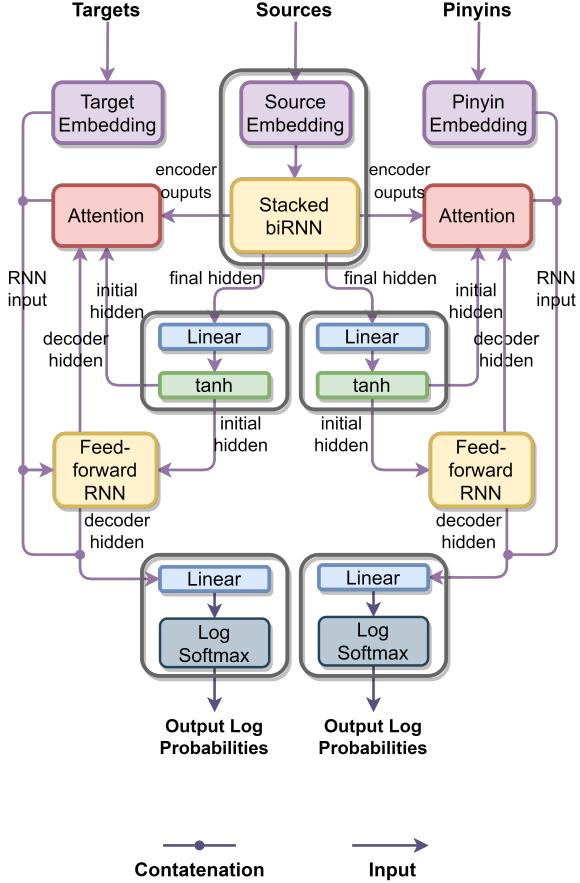


Figure 1: Visualization of the Seq2MultiSeq model. The left half illustrates the components involved in the main task and the right half is for the auxiliary task. The shared part is the encoder that consists of a source embedding layer and a stacked biRNN (top middle).

## 4 Model

Our model is intent on solving English-to-Chinese transliteration through joint supervised learning of source-to-target (main) and source-to-Pinyin (auxiliary) tasks. Training closely related tasks together can help the model to learn information that is often ignored in single-task learning, thus obtaining a better representation in the shared layers (in our case, encoder). Moreover, the auxiliary task implicitly provides the phonetic information that is not easily learned through the single main task given the characteristics of Chinese (see Section 1). Our model has a sequence-to-multiple-sequence (Seq2MultiSeq) architecture that contains a shared encoder and dual decoders. Between the encoder and decoder is a bridge layer<sup>12</sup> that transforms the

<sup>12</sup>We call it “bridge” because it connects the shared encoder to each decoder. It allows flexible choices of the hidden sizes of the encoder and decoder and serves as the intermediate “buffer” before passing the encoder final state to each decoder.

encoder’s final state into the decoder’s initial state (see Figure 1).

The encoder has an embedding layer with dropout (Hinton et al., 2012), followed by a 2-layer biLSTM (Schuster and Paliwal, 1997). The bridge layer consists of a linear layer followed by tanh activation. The shared encoder passes its final state to the main-task decoder and the auxiliary-task decoder via separate bridge layers. In each decoder, we use additive attention (Bahdanau et al., 2015) to compute the context vector (weighted sum of the encoder outputs according to the attention scores), then concatenate it with the target embedding to form the input of the subsequent 2-layer feed-forward LSTM. The prediction is made by feeding the concatenation of the LSTM’s output, the context vector and the target embedding into a linear layer followed by log-softmax.

Our model is expected to simultaneously maximize the conditional probabilities mentioned in Section 2. To achieve this goal, we use the linear combination of the main-task decoder’s loss<sup>13</sup> (negative log likelihood;  $l_y$ ) and the auxiliary-task decoder’s loss ( $l_p$ ) as the model’s objective function:

$$l_{\text{MTL}} = \lambda \cdot l_y + (1 - \lambda) \cdot l_p,$$

where the subscript MTL stands for multi-task learning and  $0 < \lambda < 1$ . Note that for  $\lambda = 0$  and  $\lambda = 1$ , it is equivalent to train on a single auxiliary task and a single main task, respectively. The whole system is implemented using the deep learning framework PyTorch (Paszke et al., 2019).<sup>14</sup>

## 5 Adaptive Evaluation Metrics

We evaluate the transliteration system using word accuracy (ACC) and its variants on the 1-best output:

$$\text{ACC} = \frac{1}{N} \sum_{(y, \hat{y})} \mathbb{I}_{\text{criterion}(\hat{y}, y)},$$

where  $N$  is the total number of test-set samples,  $\mathbb{I}_{\text{criterion}(\hat{y}, y)}$  is an indicator function with value 1 if the prediction (top candidate)  $\hat{y}$  matches the reference  $y$  under certain criterion. The simplest criterion is exact string match between  $\hat{y}$  and  $y$ . If the test set contains multiple target words for a single source word, we let indicator be 1 if the prediction matches one of the references (Chen et al., 2018).

<sup>13</sup>We use `nn.NLLLoss()` from the PyTorch library.

<sup>14</sup>Available at <https://pytorch.org/>.

Source	Target (F)	Target (M)	MED
Mona	莫娜	莫纳	1
Colina	科莉娜	科利纳	2

Table 4: Examples of a single source name with more than one target transliterations, with (F) and (M) indicating female and male, respectively.

We use ACC and ACC+ to denote the original accuracy and its variant with multiple references.

The drawback of ACC is that it may underestimate the quality of the system because it neglects the possibility of having more than one transliteration for a given source name, as is the case for English-to-Chinese transliteration. For example in Table 4, if the test set only includes Target (F) for a Source while the model predicts Target (M), ACC will mistakenly count it as wrong. Although ACC+ considers the alternatives appearing in the dataset, it is unrealistic to expect the dataset to contain all possible references. To resolve this issue, we propose a new variant of word accuracy specific to English-to-Chinese transliteration.

Based on the knowledge of a native Chinese speaker, we analyze the English-to-Chinese dataset and summarize the key observations for source names with multiple target transliterations as follows: the minimum edit distance (MED) between any two target names  $\leq 2$ , and the lengths are the same; for any two such target names, distinct characters occur in the same position, and they often indicate the *gender* of the name (see Table 4).

To use ACT in accord with the above observations, we propose the following criterion for the accuracy indicator function (we refer to it as ACC-ACT). Let subscript  $t$  denote the position of a character, then  $\mathbb{I}_{\text{criterion}(\hat{y}, y)} = 1$  if either  $\text{MED}(\hat{y}, y) = 0$  (which covers all the cases for ACC) or the following conditions are met **in order**:

1.  $\hat{y}$  and  $y$  are of the same length,  $L$ ;
2.  $\text{MED}(\hat{y}, y) \leq 2$  and distinct characters of  $\hat{y}$  and  $y$  must occur in the same position(s);
3. If  $\hat{y}_t \neq y_t$  for  $1 \leq t \leq L$ , replace  $\hat{y}_t$  by looking up the ACT and this condition will be satisfied if any of the modified  $\hat{y}$ (s) can match  $y$  exactly.

There is no guarantee that characters that are interchangeable according to ACT can replace each other in every scenario. But since we only apply

		Enc	Dec-M	Dec-A
Emb.	$h$	256	256	128
	$\delta$	0.1	0.1	0.1
RNN	$h$	512	512	128
	$\delta$	0.2	0.2	0.1

Table 5: Illustration of the model settings, where Emb. and RNN stand for the embedding layers and RNN units in each part (column) of the model,  $h$  and  $\delta$  are the hidden size and dropout value, respectively. The column names (from left to right) stand for encoder, main-task decoder and auxiliary-task decoder.

substitution on the output predictions rather than the references, we are not manipulating the test set by creating any new instance. This new metric (ACC-ACT) will ensure cases like in Table 4 are captured without requiring extra data in the test set, thus giving a more reasonable estimate of the system’s quality than both ACC and ACC+.

## 6 Experimental Setup

Recall from Section 4 that we use  $\lambda$  to denote the weighting of the two tasks we train. We set the single-main-task ( $\lambda = 1$ ) and the single-auxiliary-task ( $\lambda = 0$ ) models as the baselines, and compare the multi-task models of different weightings ( $\lambda \in \{\frac{1}{6}, \frac{1}{4}, \frac{1}{2}, \frac{2}{3}, \frac{5}{6}, \frac{8}{9}\}$ ) against them. We conduct experiments on both the NEWS and DICT datasets and select the best model for each of them to compare to the previous state of the art.

### 6.1 Model and Training Settings

The configurations of hidden sizes and dropout values of embedding layers and RNN units are presented in Table 5. The type of all RNN units is LSTM and the number of layers is set to 2. Besides the bridge layer that transforms the encoder’s final hidden state to the decoder’s initial hidden state, we add another one to carry the final cell state for using LSTM (in total, we have 4 “bridges”).

We use the Adam optimizer (Kingma and Ba, 2015) with the batch size set to 64. Evaluation of the development set is carried out on every 500 batches. We record the validation score (ACC) and decrease the learning rate (initially set to 0.003) by 90% if the score does not surpass the previous best. We pick the final model that attains the highest validation score within 100 training epochs.

For decoding in the training phase, we apply teacher forcing (Williams and Zipser, 1989) with

NEWS				DICT			
$\lambda$	Main			Auxiliary	Main		Auxiliary
	ACC	ACC+	ACC-ACT	ACC	ACC	ACC-ACT	ACC
1	0.723	0.731	0.746	NA	0.725	0.748	NA
1/6	0.666	0.672	0.688	0.698	0.728	0.750	0.744
1/4	0.734	0.743	0.751	0.755	0.725	0.747	<b>0.746</b>
1/2	0.724	0.733	0.740	0.738	0.723	0.748	0.739
2/3	0.698	0.707	0.715	0.705	0.722	0.746	0.739
5/6	<b>0.739</b>	<b>0.749</b>	<b>0.760</b>	<b>0.757</b>	<b>0.729</b>	<b>0.752</b>	<b>0.746</b>
8/9	0.670	0.679	0.686	0.705	0.722	0.746	0.734
0	NA	NA	NA	0.743	NA	NA	0.743

Table 6: Experiment results on NEWS internal test set and DICT development set, where  $\lambda = 1$  and  $\lambda = 0$  are baselines of main task and auxiliary task, respectively. Maximum score in each metric is bold.

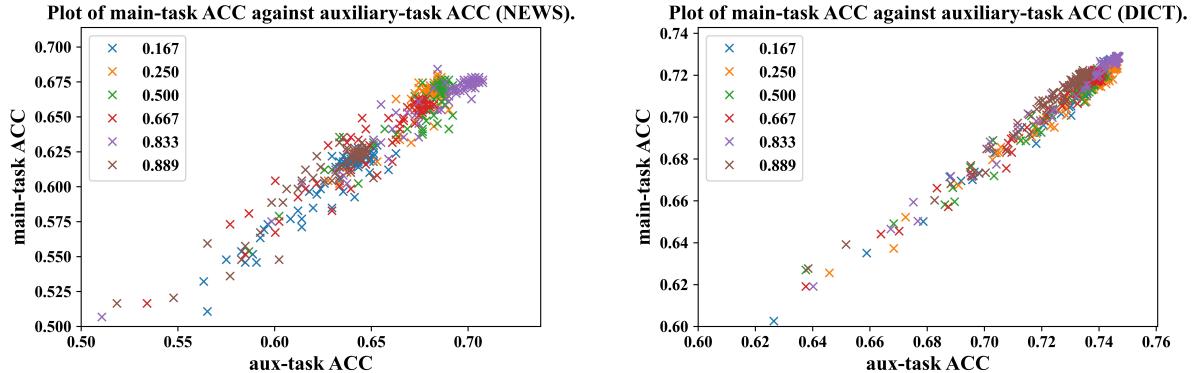


Figure 2: The plots of main-task ACC against auxiliary-task ACC on the NEWS (left) and DICT (right) development sets. Colors indicate which multi-task model (by  $\lambda$  value) the evaluation points belong to. To highlight the dense regions, we set the minimum of the x-axis to 0.5 and 0.6 for NEWS and DICT datasets, respectively.

the following empirical decay function:

$$\text{tfr} = \max\left(1 - \frac{10 + \text{epoch} \times 1.5}{50}, 0.2\right),$$

where tfr refers to the teacher forcing ratio, i.e. the probability of feeding the true reference instead of the predicted token. We use beam search decoding with beam size 10 and length normalization (Wu et al., 2016) for evaluation.

## 6.2 Evaluation

We use ACC and ACC-ACT to evaluate the performance on the main task and ACC on the auxiliary task. Note that since the only data portion we have that contains multiple references given a source word is the internal test set of NEWS data, we apply ACC+ on this particular set exclusively.

## 6.3 Model Selection

In the experiments in this section, we tune  $\lambda$  on the NEWS internal test set and DICT development set, and select the model with the highest ACC on the main task.

The experiment results in Table 6 show that  $\lambda = \frac{5}{6}$  yields the best models on both datasets. We observe a significant improvement against the baselines on NEWS while a less noticeable increase on DICT. Besides, the models are more sensitive to  $\lambda$  on NEWS than DICT (with standard deviation 0.03 and 0.003 on ACC, respectively).

Furthermore, we investigate the relationship between the main and the auxiliary tasks based on the evaluation points of the development set. In Figure 2, we observe a nearly-total positive linear correlation between the main-task ACC and auxiliary-task ACC, and this is further evident in the Pearson cor-

System	Internal Test			Official Test	
	Main			Auxiliary	Main
	ACC	ACC+	ACC-ACT	ACC	ACC+
Baseline	0.724	0.733	0.742	0.736	NA
Multi-task	0.739	0.749	0.760	0.757	0.299
BiDeep	0.731	0.739	0.746	0.740	NA
BiDeep+	NA	0.765	NA	NA	0.304

Table 7: Experiment results on the NEWS internal test (official development) set and official test set, where “Baseline” refers to the single-task model and “BiDeep+” refers to the best system [Grundkiewicz and Heafield \(2018\)](#) submitted to the NEWS workshop, and the corresponding scores are taken from their paper.

System	Main		Auxiliary
	ACC	ACC-ACT	ACC
Baseline	0.726	0.748	0.738
Multi-task	0.729	0.751	0.749
BiDeep	0.732	0.755	0.760

Table 8: Experiment results on the DICT test set, where Baseline refers to the single-task model.

User	ACC+	F-score
romang	0.3040 (1)	0.6791 (2)
Ours	0.2990 (2)	0.6799 (1)
saeednajafi	0.2820 (3)	0.6680 (3)
soumyadeep	0.2610 (4)	0.6603 (4)

Table 9: Table of the NEWS leaderboard (available at <https://competitions.codalab.org/competitions/18905#results>, accessed 19 June 2020). User “romang” refers to [Grundkiewicz and Heafield \(2018\)](#).

relation coefficients<sup>15</sup>, which are 0.982 and 0.992 for NEWS and DICT, respectively. This means the multi-task model improves the performance on both tasks simultaneously.

#### 6.4 Test-set Results and System Comparison

We submit our 1-best transliteration results on the NEWS official test set through the CodaLab link provided by the Shared Task’s Committee and we present the leaderboard partially in Table 9. Note that in addition to ACC+, the leaderboard also

<sup>15</sup>Computed by `pearsonr()` from Scipy library, which is available at: <https://docs.scipy.org/doc/scipy-0.14.0/reference/generated/scipy.stats.pearsonr.html>.

records mean F-score<sup>16</sup> on which we rank first.

We report the test-set performance of our best multi-task model on NEWS in Table 7 and DICT in Table 8, in comparison to the system built by [Grundkiewicz and Heafield \(2018\)](#). The baseline model of their work employs the RNN-based BiDeep<sup>17</sup> architecture ([Miceli Barone et al., 2017](#)) which consists of 4 bidirectional alternating stacked encoder, each with a 2-layer transition RNN cell, and 4 stacked decoders with base RNN of depth 2 and higher RNN of depth 4 ([Zhou et al., 2016; Pascanu et al., 2014; Wu et al., 2016](#)). Besides, they strengthen the model by applying layer normalization ([Ba et al., 2016](#)), skip connections ([Zhang et al., 2016](#)) and parameter tying ([Press and Wolf, 2017](#)). We reproduce their model without changing any configurations in their paper ([Grundkiewicz and Heafield, 2018](#)), and train it on both tasks separately.

In Table 7, we can see that the multi-task model performs significantly better than both the single-task baseline and the BiDeep model in all metrics on NEWS. Note that the BiDeep model we reproduce achieves the same ACC+ as reported in the work of [Grundkiewicz and Heafield \(2018\)](#) and ACC+ is the only evaluation metric used in their paper. “BiDeep+” in the third row refers to the final system they submitted to the Shared Task, on which they adopted additional NMT techniques including ensemble modeling for re-ranking and synthetic data generated from back translation ([Sennrich et al., 2017](#)). Our ACC+ score on

<sup>16</sup>The F-score metric measures the similarity between the target prediction and reference. Precision and Recall in this particular F-score are computed based on the length of the Longest Common Subsequence. See details in the NEWS whitepaper ([Chen et al., 2018](#)).

<sup>17</sup>Implemented with the Marian toolkit available at <https://marian-nmt.github.io/docs/>.

Source	Output (ST)	Output (MT)
ocallaghan	奥卡拉根	奥卡拉汉 ✓
holleran	霍尔伦	霍勒伦 ✓
ajemian	阿赫米安	阿杰米安

Table 10: Example outputs and the corresponding source words of our systems, where “ST” and “MT” refer to “single-task” and “multi-task” models. The tick symbols indicate which outputs match the references.

the anonymized official test set is 0.299 which is slightly worse than their 0.304. However, we attain a better F-score (0.6799) than them (0.6791) as shown in Table 9. Moreover, our model is of size 22M parameters, which is much smaller than their baseline BiDeep of size 133M parameters,<sup>18</sup> and we do not apply as many NMT techniques as they did. Nevertheless, on the DICT test set, there is no prominent difference among the single-task baseline, multi-task and BiDeep model, possibly because the noise pattern in the DICT dataset is not complex enough to reflect the learning ability of these models.

## 7 Discussion

In our experiments, a system has ACC-ACT>ACC+>ACC because both ACC-ACT and ACC+ consider the cases of ACC but ACC-ACT can capture more acceptable transliterations. Despite a consistent ranking given by the three metrics, ACC-ACT reveals different information from ACC and ACC+. For example, in Table 6, the model of  $\lambda = \frac{5}{6}$  outperforms  $\lambda = \frac{1}{2}$  by 0.015 and 0.016 in ACC and ACC+, respectively, but the difference is 0.020 in ACC-ACT, on the NEWS dataset. This suggests a more prominent gap between these two models. In contrast, by looking at the same two rows but on the DICT dataset, ACC-ACT indicates a smaller gap (0.004) than ACC (0.006). If we conduct experiments on another dataset, the disagreement among the metrics might be significant enough to render an inconsistent ranking.

Furthermore, we present some typical examples in which the multi-task model generates better predictions than the single-task in Table 10. In the first

example, the single-task model wrongly maps the sub-word *ghan* to 根 (emphasizing on the character *g*) while the multi-task model correctly maps *han* to 汉. The erroneous grouping of the English characters also occurs in the second example where the single-task model maps *er* to 尔 instead of more reasonably *ler* to 勒. Even in the third example where both outputs are mismatched, the multi-task model predicts the character 杰, which is closer to the source sub-word *je* than the single-task model’s 赫 in terms of pronunciation. Overall, it seems that the multi-task model can capture the source-word pronunciation better than the single-task one.

Still, the multi-task model does not consistently handle all names better than the single-task model—especially for exceptional names that do not have a regular transliteration. For instance, the name *Fyleman* is transliterated into 法伊尔曼, but the character 伊 does not have any source-word correspondence if we consider the pronunciation of the source name.

Finally, our model can be generalized to other transliteration tasks by replacing Pinyin with other phonetic representations such as IPA for English and rōmaji for Japanese. In addition, ACC-ACT can be extended to alphabetical languages by, for instance, constructing the Alternating Sub-word Table which stores lists of interchangeable subsequences. Another possible future work is to redesign the objective function by treating  $\lambda$  as a trainable parameter or including the correlation information (Papasarantopoulos et al., 2019).

## 8 Related Work

Previous work has demonstrated the effectiveness of using MTL on models through joint learning of various NLP tasks such as machine translation, syntactic and dependency parsing (Luong et al., 2016; Dong et al., 2015; Li et al., 2014). In most of this work, underlies a similar idea to create a unified training setting for several tasks by sharing the core parameters. Besides, machine transliteration has a long history of using phonetic information, for example, by mapping a phrase to its pronunciation in the source language and then convert the sound to the target word (Knight and Graehl, 1997). There is also relevant work that uses both graphemes and phonemes to various extents for transliteration, such as the correspondence-based (Oh et al., 2006) and G2P-based (Le and Sadat, 2018) approaches. Our work is inspired by the intu-

<sup>18</sup>We compute the size of our multi-task model by counting the number of trainable parameters extracted from `model.parameters()`; For the BiDeep model, we use the `numpy` package to load the model in `.npz` format and calculate the number of parameters via a simple for-loop.

itive understanding that pronunciation is essential for transliteration, and the success of incorporating phonetic information such as Pinyin (Jiang et al., 2009) and IPA (Salam et al., 2011), in the model design.

## 9 Conclusion

We argue in this paper that language-specific features should be used when solving transliteration in a neural setting, and we exemplify a way of using phonetic information as the transferred knowledge to improve a neural machine transliteration system. Our results demonstrate that the main transliteration task and the auxiliary phonetic task are indeed mutually beneficial in English-to-Chinese transliteration, and we discuss the possibility of applying this idea on other language pairs.

## Acknowledgements

We thank the anonymous reviewers for their insightful feedback. We would also like to thank Zheng Zhao, Zhijiang Guo, Waylon Li and Pinzhen Chen for their help and comments.

## References

- Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization. *ArXiv*, abs/1607.06450.
- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.
- Nancy Chen, Rafael E. Banchs, Xiangyu Duan, Min Zhang, and Haizhou Li, editors. 2018. *Proceedings of the Seventh Named Entities Workshop*. Association for Computational Linguistics, Melbourne, Australia.
- Daxiang Dong, Hua Wu, Wei He, Dianhai Yu, and Haifeng Wang. 2015. Multi-task learning for multiple language translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1723–1732, Beijing, China. Association for Computational Linguistics.
- Roman Grundkiewicz and Kenneth Heafield. 2018. Neural machine translation techniques for named entity transliteration. In *Proceedings of the Seventh Named Entities Workshop*, pages 89–94, Melbourne, Australia. Association for Computational Linguistics.
- Ulf Hermjakob, Kevin Knight, and Hal Daumé. 2008. Name translation in statistical machine translation - learning when to transliterate. In *ACL*.
- Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *ArXiv*, abs/1207.0580.
- Xue Jiang, Le Sun, and Dakun Zhang. 2009. A syllable-based name transliteration system. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration (NEWS 2009)*, pages 96–99, Suntec, Singapore. Association for Computational Linguistics.
- Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.
- Kevin Knight and Jonathan Graehl. 1997. Machine transliteration. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics, ACL '98/EACL '98*, page 128–135, USA. Association for Computational Linguistics.
- Ngoc Tan Le and Fatiha Sadat. 2018. Low-resource machine transliteration using recurrent neural networks of Asian languages. In *Proceedings of the Seventh Named Entities Workshop*, pages 95–100, Melbourne, Australia. Association for Computational Linguistics.
- Zhenghua Li, Min Zhang, Wanxiang Che, Ting Liu, and Wenliang Chen. 2014. Joint optimization for chinese pos tagging and dependency parsing. *Audio, Speech, and Language Processing, IEEE/ACM Transactions on*, 22:274–286.
- Minh-Thang Luong, Quoc V. Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. 2016. Multi-task sequence to sequence learning. *CoRR*, abs/1511.06114.
- Yuval Marton and Imed Zitouni. 2014. Transliteration normalization for information extraction and machine translation. *Journal of King Saud University - Computer and Information Sciences*, 26(4):379 – 387. Special Issue on Arabic NLP.
- Yuval Merhav and Stephen Ash. 2018. Design challenges in named entity transliteration. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 630–640, Santa Fe, New Mexico, USA. Association for Computational Linguistics.
- Antonio Valerio Miceli Barone, Jindřich Helcl, Rico Sennrich, Barry Haddow, and Alexandra Birch. 2017. Deep architectures for neural machine translation. In *Proceedings of the Second Conference on Machine Translation*, pages 99–107, Copenhagen, Denmark. Association for Computational Linguistics.

- Jong-Hoon Oh, Key-Sun Choi, and Hitoshi Isahara. 2006. A machine transliteration model based on correspondence between graphemes and phonemes. *ACM Trans. Asian Lang. Inf. Process.*, 5:185–208.
- Nikos Papasarantopoulos, Lea Frermann, Mirella Lapata, and Shay B. Cohen. 2019. Partners in crime: Multi-view sequential inference for movie understanding. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2057–2067, Hong Kong, China. Association for Computational Linguistics.
- Razvan Pascanu, Çağlar Gülcöhre, Kyunghyun Cho, and Yoshua Bengio. 2014. How to construct deep recurrent neural networks. *CoRR*, abs/1312.6026.
- Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8026–8037. Curran Associates, Inc.
- Ofir Press and Lior Wolf. 2017. Using the output embedding to improve language models. *ArXiv*, abs/1608.05859.
- Mihaela Rosca and Thomas Breuel. 2016. Sequence-to-sequence neural network models for transliteration. *CoRR*, abs/1610.09565.
- Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *CoRR*, abs/1706.05098.
- Khan Md. Anwarus Salam, Yamada Setsuo, and Tetsuro Nishino. 2011. Translating unknown words using wordnet and ipa-based-transliteration. *14th International Conference on Computer and Information Technology (ICCIT 2011)*, pages 481–486.
- Mike Schuster and Kuldip Paliwal. 1997. Bidirectional recurrent neural networks. *Signal Processing, IEEE Transactions on*, 45:2673 – 2681.
- Rico Sennrich, Orhan Firat, Kyunghyun Cho, Alexandra Birch, Barry Haddow, Julian Hitschler, Marcin Junczys-Dowmunt, Samuel Läubli, Antonio Valerio Miceli Barone, Jozef Mokry, and Maria Nädejde. 2017. Nematus: a toolkit for neural machine translation. In *Proceedings of the Software Demonstrations of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 65–68, Valencia, Spain. Association for Computational Linguistics.
- Richard Sproat, Tao Tao, and ChengXiang Zhai. 2006. Named entity transliteration with comparable corpora. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 73–80, Sydney, Australia. Association for Computational Linguistics.
- Ronald J. Williams and David Zipser. 1989. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1:270–280.
- Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Gregory S. Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google’s neural machine translation system: Bridging the gap between human and machine translation. *ArXiv*, abs/1609.08144.
- Hongbing Xing, Hua Shu, and Ping Li. 2006. The acquisition of chinese characters : Corpus analyses and connectionist simulations.
- Xinhua News Agency. 2007. *Names of the World’s Peoples: a comprehensive dictionary of names in Roman-Chinese*. China Translation & Publishing Corporation.
- Saizheng Zhang, Yuhuai Wu, Tong Che, Zhouhan Lin, Roland Memisevic, Ruslan Salakhutdinov, and Yoshua Bengio. 2016. Architectural complexity measures of recurrent neural networks. *ArXiv*, abs/1602.08210.
- Jie Zhou, Ying Cao, Xuguang Wang, Peng Li, and Wei Xu. 2016. Deep recurrent models with fast-forward connections for neural machine translation. *Transactions of the Association for Computational Linguistics*, 4:371–383.

## A Alternating Character Table in Full

Alternating Characters
莉,利,里,丽
弗,夫,芙
思,斯,丝
妮,内,娜,纳,尼
萨,沙,莎
亚,娅
玛,马,穆
琳,林
芭,巴
茜,西,锡
萝,罗
滕,坦
莱,来,勒
代,黛,戴
瓦,沃,娃
吉,姬,基
雷,蕾
薇,维,威
鲁,卢,露
塔,特
尤,于
安,阿
菲,费
纽,努
范,文
蒙,莫
查,恰
保,葆
柯,科

Table 11: The Alternating Character Table in full.

# Predicting and Using Target Length in Neural Machine Translation

Zijian Yang Yingbo Gao Weiyue Wang Hermann Ney

Human Language Technology and Pattern Recognition Group

Computer Science Department

RWTH Aachen University

D-52056 Aachen, Germany

{zyang | gao | wwang | ney}@i6.informatik.rwth-aachen.de

## Abstract

Attention-based encoder-decoder models have achieved great success in neural machine translation tasks. However, the lengths of the target sequences are not explicitly predicted in these models. This work proposes length prediction as an auxiliary task and set up a sub-network to obtain the length information from the encoder. Experimental results show that the length prediction sub-network brings improvements over the strong baseline system and that the predicted length can be used as an alternative to length normalization during decoding.

## 1 Introduction

In recent years, neural network (NN) models have achieved great improvements in machine translation (MT) tasks. Sutskever et al. (2014) introduced the encoder-decoder network, Bahdanau et al. (2015) developed the attention-based architecture, and Vaswani et al. (2017) proposed the transformer model with self-attentions, which delivers state-of-the-art performances.

Despite the success achieved in neural machine translation (NMT), current NMT systems do not model the length of the output explicitly, and thus various length normalization approaches are often used in decoding. Length normalization is a common technique used in the beam search of NMT systems to enable a fair comparison of partial hypotheses with different lengths. Without any form of length normalization, regular beam searches will prefer shorter hypotheses to longer ones on average, as a negative logarithmic probability is added at each step, resulting in lower (more negative) scores for longer sentences. The simplest way is to normalize the score of the current partial hypothesis ( $e_1^i$ ) by its length ( $|i|$ ):

$$s(e_1^i, f_1^J) = \frac{\log p(e_1^i | f_1^J)}{|i|} \quad (1)$$

where  $f_1^J$  is the source sequence. To use a softer approach, the denominator  $|i|$  can also be raised to the power of a number between 0 and 1 or replaced by more complex functions, as proposed in Wu et al. (2016). Moreover, a constant word reward is used in He et al. (2016) as an alternative to length normalization. All of these approaches tackle the length problem in decoding, and all NMT systems use at least one of them to ensure the performance.

In addition to investigating various types of length normalization, their rationality is rarely explored. Although length normalization appears to be simple and effective, it is still an additional technique to help a “weak” machine translation model that cannot handle the hypothesis length properly. In this work it is proposed to model the target length using the neural network itself in a multi-task learning way. The estimated length information can either be implicitly included in the network to “guide” translation, or it can be used explicitly as an alternative to length normalization during decoding. The experimental results on various datasets show that the proposed system achieves improvements compared to the baseline model and the predicted length can easily be used to replace the length normalization.

## 2 Related Work

Multi-task learning is an important training strategy that aims to improve the generalization performance of the main task with some other related tasks (Luong et al., 2016; Martínez Alonso and Plank, 2017). With regard to deep learning, multi-task learning is applied successfully in many areas, such as natural language processing (Liu et al., 2015), computer vision (Donahue et al., 2014), and speech processing (Heigold et al., 2013). In this work, the prediction of the target length while generating translation hypotheses can be seen as a

multi-task learning application.

Murray and Chiang (2018) and Stahlberg and Byrne (2019) attribute the fact that beam search prefers shorter candidates due to the local normalization of NMT. To address this problem, in addition to the standard length normalization technique, Wu et al. (2016) propose a more complicated correction with a hyperparameter that can be adjusted for different language pairs. In He et al. (2016), a word reward function is proposed that simulates the coverage vector in statistical machine translation so that the decoder prefers a long translation. Huang et al. (2017) and Yang et al. (2018) suggest variations of this reward that provide better guarantees during search. There are also works on target vocabulary prediction in the encoder-decoder model that implicitly predicts the target length (Weng et al., 2017; Suzuki and Nagata, 2017). In our work, the target length is explicitly modeled by the neural network itself, which indicates that the entire system relies more on statistics rather than heuristics.

### 3 Neural Length Model

To predict the target length based on the standard transformer architecture (Vaswani et al., 2017), we build a multi-layer sub-network that only requires information from the source sequence (or the encoder). In this work the length prediction task is considered as a classification task for different lengths. Other methods, such as directly generating a real number, binarizing the length, or performing multiple binary classification tasks, are also being tested, but the classification method performs best.

#### 3.1 Modeling

We predict the length of the target sequence by a classifier in the range of  $[0, 200]$ , the input of which is a single vector without time dimension, which is extracted from the encoder. To obtain this vector, we first concatenate the encoder output and the embedding of the source tokens, followed by a linear layer with an activation function to map the vectors to the same dimension as the original encoder output. Then we set the length of the concatenated vectors to 200 by clipping or zero padding, in order to have a fixed length of time dimension, which could be compressed to a single vector by convolution and max-pooling. Then, the vectors run through a convolutional layer with an activation function and a max-pooling layer. A linear layer is

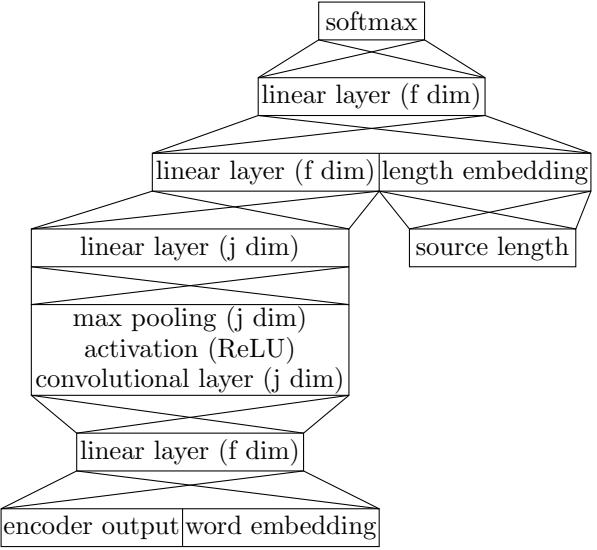


Figure 1: The architecture of the length prediction sub-network.

then used to project the max-pooled vector into a single vector.

We also embed the length of source sequence into a 201 dimension vector with a length embedding matrix, which is initialized by the empirical distribution of the length. This length embedding is then concatenated with the output logit of the length prediction sub-network. Again, this concatenated vector is projected through a linear layer onto a vector  $s$  with 201 dimensions. Finally, the length distribution  $q_l$  is given by a softmax over  $s$ . And the predicted length  $l_{\text{pred}}$  is the  $l$  with the highest probability. The complete structure of the proposed length prediction sub-network is illustrated in Figure 1.

When we train the model with the translation and length prediction tasks jointly, the gradient of the length model will propagate to the translation model (referred to as no-connection in this paper). Thus, these two models will influence each other during multi-task training. In addition, the translation model could benefit from concatenating the length prediction output vector  $s$  to the outputs of each decoder layer (referred to as cross-concat in this paper). After the vector is concatenated, a linear projection is run through to maintain the feature dimension of the vector as the original one, so that it can be used without modifying the rest of the original transformer model. Here we detach  $s$  from the backpropagation graph so that the length prediction is not affected by this connection. In this method, we think that with the concatenation, the

length information could be passed to the decoder and used implicitly.

### 3.2 Training

During training, Kullback–Leibler (KL) divergence ([Kullback and Leibler, 1951](#)) is used as the loss of the length prediction task:

$$\text{Loss}_{\text{length}} = D_{\text{KL}}(P||Q) = \sum_l p_l \log \frac{p_l}{q_l} \quad (2)$$

where  $q_l$  is the probability from model output. Suppose  $l_{\text{target}}$  is the actual length of the target sequence,  $p_l$  is the target distribution given by a Gaussian function added with a neighborhood reward  $d(l, l_{\text{target}})$ . Formally,  $p_l$  is given as:

$$p_l = \frac{a_l}{\sum_{l'} a_{l'}} \quad (3)$$

where

$$a_l = \exp \left( - \left( \frac{l - l_{\text{target}}}{\sigma} \right)^2 \right) + d(l, l_{\text{target}}) \quad (4)$$

where

$$d(l, l_{\text{target}}) = \begin{cases} 1 & \text{if } l = l_{\text{target}} \\ 0.1 & \text{if } |l - l_{\text{target}}| = 1 \\ 0 & \text{others} \end{cases} \quad (5)$$

here  $\sigma$  is a constant and is used to control the shape of the distribution. In contrast to cross entropy with label smoothing, in which there is only one true label with a high probability and others are treated equally, the probability  $p_l$  becomes smaller if  $l$  is further away from  $l_{\text{target}}$ , which creates the desired relationship between each class in the classifier.

We use cross entropy with label smoothing as the training loss for the translation task. We linearly combine the translation loss with the length loss, so that the training loss is given by

$$\text{Loss}_{\text{all}} = \lambda_1 \text{Loss}_{\text{translation}} + \lambda_2 \text{Loss}_{\text{length}} \quad (6)$$

### 3.3 Decoding

Besides using the length information implicitly (as the two methods mentioned above), we can also guide the decoding step with the length prediction explicitly. With the help of the length prediction, we have a mathematically reasonable control of the output length in comparison to the length normalization in beam search. Since the predicted target length cannot be 100% accurate and a source

sentence can have multiple possible translations of different lengths, we control the length of the inference by penalizing the score (logarithmic probability) of the end-of-sentence (EOS) token during beam search, rather than forcing the length of the inference to match the predicted length. More specifically, if the length of the hypothesis is shorter than the predicted length, the EOS token score is penalized; if the hypothesis is longer than the predicted length, the EOS token score is rewarded to facilitate the selection of the EOS token in beam search to finalize the hypothesis. A logarithmic linear penalty is introduced, which is added to the score of EOS token at each time step during beam search:

$$P = \alpha \log \frac{L_{\text{hyp}}}{L_{\text{pred}}} \quad (7)$$

where  $L_{\text{hyp}}$  is the length of the hypothesis,  $L_{\text{pred}}$  is the predicted length of the target sentence, and  $\alpha$  is a hyperparameter to control the penalty.

## 4 Experiments

### 4.1 Experimental Setup

We first conduct experiments on a relatively small dataset, IWSLT2014 German→English (160k sentence pairs) ([Cettolo et al., 2014](#)), to tune hyperparameters and analyze the performance. Then we train our model on other four different language pairs, which are Spanish-English (es-en), Italian-English (it-en), Dutch-English (nl-en) and Romanian-English (ro-en). At last, the experiments are carried out on the WMT ([Barrault et al., 2019](#)) German↔English (4M sentence pairs) datasets in order to compare our system with the baseline model. All datasets used in this work are preprocessed by *fairseq*<sup>1</sup> ([Ott et al., 2019](#)). Data statistics can be found in Table 1.

data set	language pair	number of sentence pairs		
		train	valid	test
IWSLT	de-en	160k	7.3k	6.8k
	es-en	169k	7.7k	5.6k
	it-en	167k	7.6k	6.6k
	nl-en	154k	7.0k	5.4k
	ro-en	168k	7.6k	5.6k
WMT	en↔de	4.5M	3.0k	3.0k

Table 1: Data statistics of IWSLT and WMT datasets.

We employ the transformer base architecture ([Vaswani et al., 2017](#)) as the baseline model and

<sup>1</sup><https://github.com/pytorch/fairseq>

this work is implemented in *fairseq*. All model hyperparameters of the baseline model for IWSLT match the settings in *fairseq*. For the WMT experiments, the settings are the same as for the original base transformer model. The sub-network used for the length prediction only increases the number of free parameters by less than 10%, the influence on the training and decoding speed is also marginal. Experimental performance is measured using BLEU (Papineni et al., 2002; Post, 2018) and CHARACTER (Wang et al., 2016) (CTER) metrics.

## 4.2 Experimental Results

For the length prediction task, the inference length does not have to correspond exactly to the reference length, since there can be multiple correct translations with different lengths. Therefore, we consider the predictions that fulfill  $|l_{\text{predict}} - l_{\text{target}}|/l_{\text{target}} \leq T$  to be accurate, where  $T$  is a threshold.

$\lambda_1$	$\lambda_2$	model	acc. [%]	BLEU[%]
1	0	baseline	-	34.8
0	1	length model	83.4	-
1	1	no-connection	86.7	35.3
		cross-concat	86.1	35.3

Table 2: Accuracy rate and BLEU scores of the proposed system with the length model on the IWSLT German→English task. The accuracy of the length prediction task is reported on the validation dataset.

language pair	es-en	it-en	nl-en	ro-en
baseline	41.2	32.6	37.8	38.4
no-connection	41.3	32.8	37.8	38.8
cross-concat	41.3	32.7	38.3	38.7

Table 3: Performance (in BLEU[%] scores) using different methods for different language pairs.

Table 2 shows the experiments carried out with the standard translation model ( $\lambda_1 = 1$  and  $\lambda_2 = 0$ ), the pure length model ( $\lambda_1 = 0$  and  $\lambda_2 = 1$ ) and the combination of the two models ( $\lambda_1 = 1$  and  $\lambda_2 = 1$ ). For the accuracy, here we choose the threshold  $T = 20\%$ . It is observed that the joint training of the two models performs better for both the translation and the length prediction task. Due to the multi-task learning, although the translation task does not explicitly influence the length prediction, it helps to bring model parameters to a better local optimum.

We use no-connection, cross-concat model to

train on other language pairs with the same hyperparameters as on IWSLT de-en to test the performance, as shown in Table 3. For cross-concat, the BLEU score of the nl-en system is improved by 0.5%. For other language pairs, the results of two methods are almost the same, all of which are better than the baseline.

Figure 2 shows the relative differences  $\Delta_l$  between the predicted and actual lengths for different target sequence lengths. When  $l_{\text{target}}$  is between about 10 and 40, the prediction is pretty good: for most  $l_{\text{target}}$  in  $[10, 40]$ ,  $\Delta_l$  is less than 15%. When  $l_{\text{target}}$  is in  $[40, 100]$ , the prediction becomes worse, but most of them are still less than 20%. After 100, the prediction is pretty bad. There are two reasons for this: first, the length of most target sequences in training data is between 10 and 40, so the model does not often see the cases that the sequence is too long; second, there are very few long sequences in the validation dataset, so the results for these points lack statistical meaning.

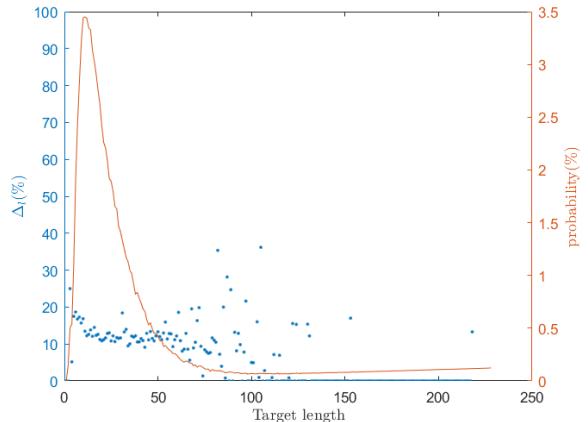


Figure 2: The left  $y$ -axis shows the relative difference between the target and the predicted length and the right  $y$ -axis is for the empirical distribution of  $l_{\text{target}}$ .

Table 4 shows the comparison between the proposed approach and the baseline model. Here we set  $\alpha = 10$  according to the experiments that are carried out on the IWSLT dataset. The additional sub-network for length prediction improves the BLEU score by up to 0.9% over the strong baseline model. Moreover, the predicted length successfully serves as an alternative to length normalization. Regardless of whether the inference tends to be longer or shorter than the reference, the ratio when using the predicted length is slightly better than using the length normalization, which shows better control of the length.

model		English→German newstest2014			German→English newstest2017		
		BLEU[%] ↑	CTER[%] ↓	len. ratio	BLEU[%] ↑	CTER[%] ↓	len. ratio
baseline		27.3	45.8	1.024	33.0	41.8	<b>0.974</b>
+ len. model	no-connection	<b>27.6</b>	<b>45.5</b>	1.020	33.4	41.5	0.972
	cross-concat	27.4	45.7	1.024	33.4	41.3	0.970
- len. norm.	no-connection	<b>27.6</b>	45.6	<b>1.018</b>	<b>33.9</b>	<b>40.9</b>	0.973
	cross-concat	27.3	45.8	1.021	33.7	41.2	<b>0.974</b>

Table 4: Comparison between the proposed system and the baseline model. “+ len. model” indicates that the length prediction sub-network is added to the baseline architecture. “- len. normalization” denotes that the predicted length is used during decoding as an alternative to the length normalization as described in Section 3.3. “len. ratio” gives the length ratio between the hypothesis length and the reference length: the closer to 1, the better.

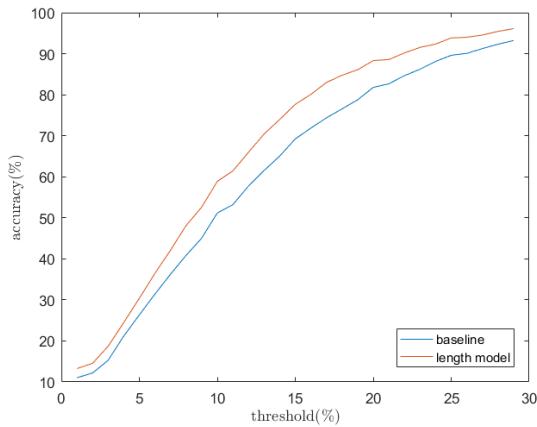


Figure 3: Length prediction model outperforms the baseline model in length prediction test accuracy.

Figure 3 shows the relationship between the length prediction accuracy of the baseline model and the length prediction model, and the threshold  $T$  for calculating accuracy. Since the transformer baseline model does not predict the target length, the length prediction of baseline is obtained from the average ratio of source sentence length to target sentence length. For the length prediction task, the accuracy of our model is always better than the baseline, which indicates that on WMT data, our model can still predict the target length well.

## 5 Conclusion

In this paper, we propose a length prediction sub-network based on the transformer architecture, and a method of using the length prediction information on the decoder side, namely cross-concat. In decoding, we use the predicted length to calculate a logarithmic linear penalty in the beam search in order to replace the length normalization. Experimental results show that the sub-network can

predict target length well and further improve translation quality. In addition, the predicted length can be used to replace the length normalization with a better and more mathematically explainable control of the output length. For future work, the use of length prediction in positional encoding (Lakew et al., 2019; Takase and Okazaki, 2019) and non-autoregressive (or partially autoregressive) NMT (Gu et al., 2017; Lee et al., 2018; Stern et al., 2019) could be further investigated.

## Acknowledgements



This work has received funding from the European Research Council (ERC) (under the European Union’s Horizon 2020 research and innovation programme, grant agreement No 694537, project “SEQCLAS”) and the Deutsche Forschungsgemeinschaft (DFG; grant agreement NE 572/8-1, project “CoreTec”). The GPU computing cluster was supported by DFG (Deutsche Forschungsgemeinschaft) under grant INST 222/1168-1 FUGG.

## References

- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *3rd International Conference on Learning Representations (ICLR)*, San Diego, CA, USA.
- Loïc Barrault, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Müller, Santanu Pal, Matt Post, and Marcos Zampieri. 2019. Findings of the 2019 conference on machine translation (WMT19). In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared*

*Task Papers, Day 1*), pages 1–61, Florence, Italy. Association for Computational Linguistics.

Mauro Cettolo, an Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. 2014. Report on the 11th iwslt evaluation campaign, iwslt 2014. In *Proceedings of the 11th International Workshop on Spoken Language Translation (IWSLT)*, Lake Tahoe, US.

Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. 2014. Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*, pages 647–655.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O. K. Li, and Richard Socher. 2017. Non-autoregressive neural machine translation. *CoRR*, abs/1711.02281.

Wei He, Zhongjun He, Hua Wu, and Haifeng Wang. 2016. Improved neural machine translation with SMT features. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12–17, 2016, Phoenix, Arizona, USA*, pages 151–157. AAAI Press.

Georg Heigold, Vincent Vanhoucke, Alan Senior, Patrick Nguyen, Marc’Aurelio Ranzato, Matthieu Devin, and Jeffrey Dean. 2013. Multilingual acoustic models using distributed deep neural networks. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8619–8623. IEEE.

Liang Huang, Kai Zhao, and Mingbo Ma. 2017. When to finish? optimal beam search for neural text generation (modulo beam size). In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2134–2139, Copenhagen, Denmark. Association for Computational Linguistics.

S. Kullback and R. A. Leibler. 1951. On information and sufficiency. *Ann. Math. Statist.*, 22(1):79–86.

Surafel Melaku Lakew, Mattia Antonino Di Gangi, and Marcello Federico. 2019. Controlling the output length of neural machine translation. *CoRR*, abs/1910.10408.

Jason Lee, Elman Mansimov, and Kyunghyun Cho. 2018. Deterministic non-autoregressive neural sequence modeling by iterative refinement. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1173–1182, Brussels, Belgium. Association for Computational Linguistics.

Xiaodong Liu, Jianfeng Gao, Xiaodong He, Li Deng, Kevin Duh, and Ye-yi Wang. 2015. Representation learning using multi-task deep neural networks for semantic classification and information retrieval. In *Proceedings of the 2015 Conference of the North*

*American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 912–921, Denver, Colorado. Association for Computational Linguistics.

Minh-Thang Luong, Quoc V. Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. 2016. Multi-task sequence to sequence learning. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Héctor Martínez Alonso and Barbara Plank. 2017. When is multitask learning effective? semantic sequence prediction under varying data conditions. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 44–53, Valencia, Spain. Association for Computational Linguistics.

Kenton Murray and David Chiang. 2018. Correcting length bias in neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 212–223, Belgium, Brussels. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Felix Stahlberg and Bill Byrne. 2019. On NMT search errors and model errors: Cat got your tongue? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3356–3362, Hong Kong, China. Association for Computational Linguistics.

Mitchell Stern, William Chan, Jamie Kiros, and Jakob Uszkoreit. 2019. Insertion transformer: Flexible sequence generation via insertion operations. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019,*

*Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 5976–5985. PMLR.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Jun Suzuki and Masaaki Nagata. 2017. **Cutting-off redundant repeating generations for neural abstractive summarization**. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 291–297, Valencia, Spain. Association for Computational Linguistics.

Sho Takase and Naoaki Okazaki. 2019. **Positional encoding to control output sequence length**. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3999–4004, Minneapolis, Minnesota. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Weiyue Wang, Jan-Thorsten Peter, Hendrik Rosendahl, and Hermann Ney. 2016. **CharacTer: Translation edit rate on character level**. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 505–510, Berlin, Germany. Association for Computational Linguistics.

Rongxiang Weng, Shujian Huang, Zaixiang Zheng, Xinyu Dai, and Jiajun Chen. 2017. **Neural machine translation with word predictions**. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 136–145, Copenhagen, Denmark. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. **Google’s neural machine translation system: Bridging the gap between human and machine translation**. *CoRR*, abs/1609.08144.

Yilin Yang, Liang Huang, and Mingbo Ma. 2018. **Breaking the beam search curse: A study of (re-)scoring methods and stopping criteria for neural machine translation**. In *Proceedings of the 2018*

*Conference on Empirical Methods in Natural Language Processing*, pages 3054–3059, Brussels, Belgium. Association for Computational Linguistics.

# Grounded PCFG Induction with Images

Lifeng Jin and William Schuler

Department of Linguistics

The Ohio State University, Columbus, OH, USA

{jin, schuler}@ling.osu.edu

## Abstract

Recent work in unsupervised parsing has tried to incorporate visual information into learning, but results suggest that these models need linguistic bias to compete against models that only rely on text. This work proposes grammar induction models which use visual information from images for labeled parsing, and achieve state-of-the-art results on grounded grammar induction on several languages. Results indicate that visual information is especially helpful in languages where high frequency words are more broadly distributed. Comparison between models with and without visual information shows that the grounded models are able to use visual information for proposing noun phrases, gathering useful information from images for unknown words, and achieving better performance at prepositional phrase attachment prediction.<sup>1</sup>

## 1 Introduction

Recent grammar induction models are able to produce accurate grammars and labeled parses with raw text only (Jin et al., 2018b, 2019; Kim et al., 2019b,a; Drozdov et al., 2019), providing evidence against the poverty of the stimulus argument (Chomsky, 1965), and showing that many linguistic distinctions like lexical and phrasal categories can be directly induced from raw text statistics. However, as computational-level models of human syntax acquisition, they lack semantic, pragmatic and environmental information which human learners seem to use (Gleitman, 1990; Pinker and MacWhinney, 1987; Tomasello, 2003).

This paper proposes novel grounded neural-network-based models of grammar induction which take into account information extracted from images in learning. Performance comparisons show



(a) friend as companion

(b) friend as condiment

Figure 1: Examples of disambiguating information provided by images for the prepositional phrase attachment of the sentence *Mary eats spaghetti with a friend* (Gokcen et al., 2018).

that the proposed models achieve state-of-the-art results on multilingual induction datasets, even without help from linguistic knowledge or pretrained image encoders. Experiments show several specific benefits attributable to the use of visual information in induction. First, as a proxy to semantics, the co-occurrences between objects in images and referring words and expressions, such as the word *spaghetti* and the plate of spaghetti in Figure 1,<sup>2</sup> provide clues to the induction model about the syntactic categories of such linguistic units, which may complement distributional cues from word collocation which normal grammar inducers rely on solely for induction. Also, pictures may help disambiguate different syntactic relations: induction models are not able to resolve many prepositional phrase attachment ambiguities with only text — for example in Figure 1, there is little information in the text of *Mary eats spaghetti with a friend* for the induction models to induce a high attachment structure where *a friend* is a companion — and images may provide information to resolve these ambiguities. Finally, images may provide grounding information for unknown words when their syntactic properties are not clearly indicated by sentential context.

<sup>1</sup>The system implementation and translated datasets used in this work can be found at <https://github.com/lifengjin/imagepcfg>.

<sup>2</sup><https://github.com/ajdagokcen/madlyambiguous-repo>

## 2 Related work

Existing unsupervised PCFG inducers exploit naturally-occurring cognitive and developmental constraints, such as punctuation as a proxy to prosody (Seginer, 2007), human memory constraints (Noji and Johnson, 2016; Shain et al., 2016; Jin et al., 2018b), and morphology (Jin and Schuler, 2019), to regulate the posterior of grammars which are known to be extremely multimodal (Johnson et al., 2007). Models in Shi et al. (2019) also match embeddings of word spans to encoded images to induce unlabeled hierarchical structures with a concreteness measure (Hill et al., 2014; Hill and Kophon, 2014). Additionally, visual information is observed to provide grounding for words describing concrete objects, helping to identify and categorize such words. This hypothesis is termed ‘noun bias’ in language acquisition (Gentner, 1982, 2006; Waxman et al., 2013), through which the early acquisition of nouns is attributed to nouns referring to observable objects. However, the models in Shi et al. (2019) also rely on language-specific branching bias to outperform other text-based models, and images are encoded by pretrained object classifiers trained with large datasets, with no ablation to show the benefit of visual information for unsupervised parsing. Visual information has also been used for joint training of prepositional phrase attachment models (Christie et al., 2016) suggesting that visual information may contain semantic information to help disambiguate prepositional phrase attachment.

## 3 Grounded Grammar Induction Model

The full grounded grammar induction model used in these experiments, ImagePCFG, consists of two parts: a word-based PCFG induction model and a vision model, as shown in Figure 2. The two parts have their own objective functions. The PCFG induction model, called NoImagePCFG when trained by itself, can be trained by maximizing the marginal probability  $P(\sigma)$  of sentences  $\sigma$ . This part functions similarly to previously proposed PCFG induction models (Jin et al., 2018a; Kim et al., 2019a) where a PCFG is induced through maximization of the data likelihood of the training corpus marginalized over latent syntactic trees.

The image encoder-decoder network in the vision model is trained to reconstruct the original image after passing through an information bottleneck. The latent encoding from the image encoder may be seen as a compressed representation of vi-

sual information in the image, some of which is semantic, relating to objects in the image. We hypothesize that semantic information can be helpful in syntax induction, potentially through helping three tasks mentioned above.

In contrast to the full model where the encoded visual representations are trained from scratch, the ImagePrePCFG model uses image embeddings encoded by pretrained image classifiers with parameters fixed during induction training. We hypothesize that pretrained image classifiers may provide useful information about objects in an image, but for grammar induction it is better to allow the inducer to decide which kind of information may help induction.

The two parts are connected through a syntactic-visual loss function connecting a syntactic sentence embedding projected from word embeddings and an image embedding. We hypothesize that visual information in the encoded images may help constrain the search space of syntactic embeddings of words with supporting evidence of lexical attributes such as concreteness for nouns or correlating adjectives with properties of objects.<sup>3</sup>

### 3.1 Induction model

The PCFG induction model is factored into three submodels: a nonterminal expansion model, a terminal expansion model and a split model, which distinguishes terminal and nonterminal expansions. The binary-branching non-terminal expansion rule probabilities,<sup>4</sup> and unary-branching terminal expansion rule probabilities in a factored Chomsky-normal-form PCFG can be parameterized with these three submodels. Given a tree as a set  $\tau$  of nodes  $\eta$  undergoing non-terminal expansions  $c_\eta \rightarrow c_{\eta 1} c_{\eta 2}$  (where  $\eta \in \{1, 2\}^*$  is a Gorn address specifying a path of left or right branches from the root), and a set  $\tau'$  of nodes  $\eta$  undergoing terminal expansions  $c_\eta \rightarrow w_\eta$  (where  $w_\eta$  is the word at node  $\eta$ ) in a parse of sentence  $\sigma$ , the marginal

<sup>3</sup>The syntactic nature of word embeddings indicates that any lexical-specific semantic information in these embeddings may be abstract, which is generally not sufficient for visual reconstruction. Experiments with syntactic embeddings show that it is difficult to extract semantic information from them and present visually.

<sup>4</sup>These include the expansion rules generating the top node in the tree.

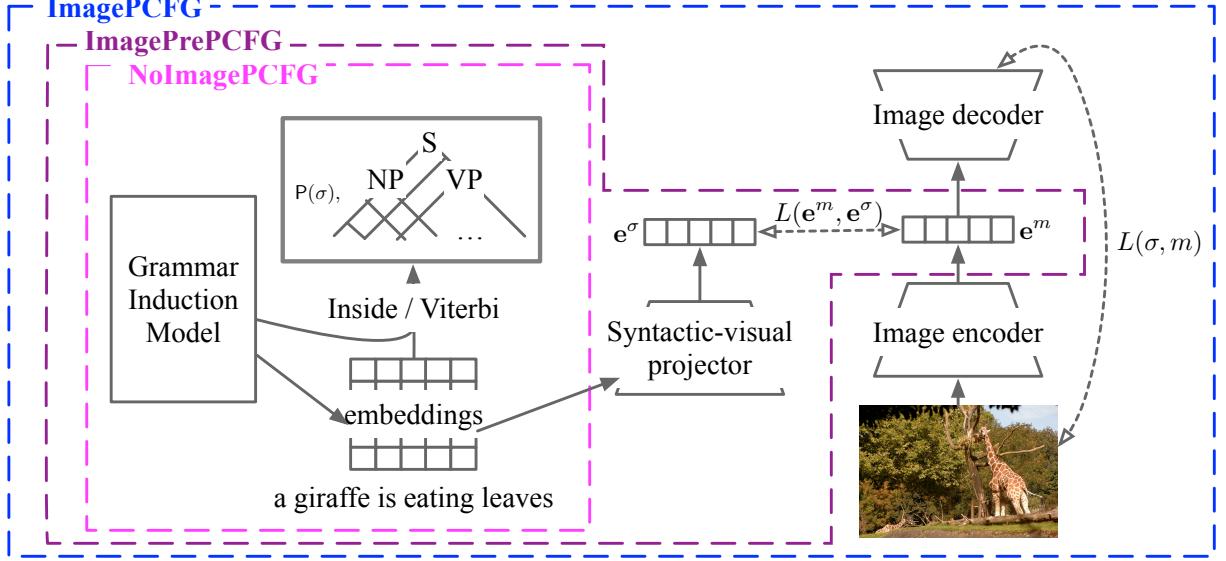


Figure 2: Different configurations of PCFG induction models: the model without vision (NoImagePCFG), the model with a pretrained image encoder (ImagePrePCFG) and the model with images (ImagePCFG.)

probability of  $\sigma$  can be computed as:

$$\mathsf{P}(\sigma) = \sum_{\tau, \tau'} \prod_{\eta \in \tau} \mathsf{P}(c_\eta \rightarrow c_{\eta 1} c_{\eta 2}) \cdot \prod_{\eta \in \tau'} \mathsf{P}(c_\eta \rightarrow w_\eta) \quad (1)$$

We first define a set of Bernoulli distributions that distribute probability mass between terminal and nonterminal rules, so that the lexical expansion model can be tied to the image model (see Section 4.2):

$$\mathsf{P}(\text{Term} | c_\eta) = \underset{\{0,1\}}{\text{softmax}}(\text{ReLU}(\mathbf{W}_{\text{spl}} \mathbf{x}_{B,c_\eta} + \mathbf{b}_{\text{spl}})), \quad (2)$$

where  $c_\eta$  is a non-terminal category,  $\mathbf{W}_{\text{spl}} \in \mathbb{R}^{2 \times h}$  and  $\mathbf{b}_{\text{spl}} \in \mathbb{R}^2$  are model parameters for hidden vectors of size  $h$ , and  $\mathbf{x}_{B,c_\eta} \in \mathbb{R}^h$  the result of a multilayered residual network (Kim et al., 2019a). The residual network consists of  $B$  architecturally identical residual blocks. For an input vector  $\mathbf{x}_{b-1,c}$  each residual block  $b$  performs the following computation:

$$\mathbf{x}_{b,c} = \text{ReLU}(\mathbf{W}_b \text{ReLU}(\mathbf{W}'_b \mathbf{x}_{b-1,c} + \mathbf{b}'_b) + \mathbf{b}_b) + \mathbf{x}_{b-1,c}, \quad (3)$$

with base case:

$$\mathbf{x}_{0,c} = \text{ReLU}(\mathbf{W}_0 \mathbf{E} \delta_c + \mathbf{b}_0) \quad (4)$$

where  $\delta_c$  is a Kronecker delta function – a vector with value one at index  $c$  and zeros everywhere else – and  $\mathbf{E} \in \mathbb{R}^{d \times C}$  is an embedding matrix for each

nonterminal category  $c$  with embedding size  $d$ , and  $\mathbf{W}_0 \in \mathbb{R}^{h \times d}$ ,  $\mathbf{W}_b, \mathbf{W}'_b \in \mathbb{R}^{h \times h}$  and  $\mathbf{b}_0, \mathbf{b}_b, \mathbf{b}'_b \in \mathbb{R}^h$  are model parameters with latent representations of size  $h$ .  $B$  is set to 2 in all models following Kim et al. (2019a). Binary-branching non-terminal expansion rule probabilities for each non-terminal category  $c_\eta$  and left and right children  $c_{\eta 1} c_{\eta 2}$  are defined as:

$$\mathsf{P}(c_\eta \rightarrow c_{\eta 1} c_{\eta 2}) = \mathsf{P}(\text{Term}=0 | c_\eta) \cdot \underset{c_{\eta 1}, c_{\eta 2}}{\text{softmax}}(\mathbf{W}_{\text{nont}} \mathbf{E} \delta_{c_\eta} + \mathbf{b}_{\text{nont}}), \quad (5)$$

where  $\mathbf{W}_{\text{nont}} \in \mathbb{R}^{C^2 \times d}$  and  $\mathbf{b}_{\text{nont}} \in \mathbb{R}^{C^2}$  are parameters of the model.

The lexical unary-expansion rule probabilities for a preterminal category  $c_\eta$  and a word  $w_\eta$  at node  $\eta$  are defined as:

$$\mathsf{P}(c_\eta \rightarrow w_\eta) = \mathsf{P}(\text{Term}=1 | c_\eta) \cdot \frac{\exp(n_{c_\eta, w_\eta})}{\sum_w \exp(n_{c_\eta, w})} \quad (6)$$

$$n_{c,w} = \text{ReLU}(\mathbf{w}_{\text{lex}}^\top \mathbf{n}_{B,c,w} + b_{\text{lex}}) \quad (7)$$

where  $w$  is the generated word type, and  $\mathbf{w}_{\text{lex}} \in \mathbb{R}^h$  and  $b_{\text{lex}} \in \mathbb{R}$  are model parameters. Similarly,

$$\mathbf{n}_{b,c,w} = \text{ReLU}(\mathbf{W}_b'' \text{ReLU}(\mathbf{W}_b''' \mathbf{n}_{b-1,c,w} + \mathbf{b}_b''') + \mathbf{b}_b'') + \mathbf{n}_{b-1,c,w}, \quad (8)$$

with base case:

$$\mathbf{n}_{0,c,w} = \text{ReLU}(\mathbf{W}'_0 \begin{bmatrix} \mathbf{E} \delta_c \\ \mathbf{L} \delta_w \end{bmatrix} + \mathbf{b}'_0) \quad (9)$$

where  $\mathbf{W}'_0 \in \mathbb{R}^{h \times 2d}$ ,  $\mathbf{W}'_b, \mathbf{W}''_b \in \mathbb{R}^{h \times h}$  and  $\mathbf{b}_0, \mathbf{b}''_b, \mathbf{b}'''_b \in \mathbb{R}^h$  are model parameters for latent representations of size  $h$ .  $\mathbf{L}$  is a matrix of syntactic word embeddings for all words in vocabulary.

## 4 Vision model

The vision model consists of an image encoder-decoder network and a syntactic-visual projector. The image encoder-decoder network encodes an image into an image embedding and then decodes that back into the original image. This reconstruction constrains the information in the image embedding to be closely representative of the original image. The syntactic-visual projector projects word embeddings used in the calculation of lexical expansion probabilities into the space of image embeddings, building a connection between the space of syntactic information and the space of visual information.

### 4.1 The image encoder-decoder network

The image encoder employs a ResNet18 architecture (He et al., 2016) which encodes an image with 3 channels into a single vector. The encoder consists of four blocks of residual convolutional networks. The image decoder decodes an image from a visual vector generated by the image encoder. The image decoder used in the joint model is the image generator from DCGAN (Radford et al., 2016), where a series of transposed convolutions and batch normalizations attempts to recover an image from an image embedding.<sup>5</sup>

### 4.2 The syntactic-visual projector

The projector model is a CNN-based neural network which takes a concatenated sentence embedding matrix  $\mathbf{M}^\sigma \in \mathbb{R}^{|\sigma| \times d}$  as input, where embeddings in  $\mathbf{M}^\sigma$  are taken from  $\mathbf{L}$ , and returns the syntactic-visual embedding  $\mathbf{e}^\sigma$ . The  $j$ th full length-wise convolutional kernel is defined as a matrix  $\mathbf{K}_j \in \mathbb{R}^{u_j \times k_j d}$  which slides across the sentence matrix  $\mathbf{M}$  to produce a feature map, where  $u_j$  is the number of channels in the kernel,  $k_j$  is the width of the kernel, and  $d$  is the height of the kernel which is equal to the size of the syntactic word embeddings. Because the kernel is as high as the embeddings, it produces one vector of length  $u_j$  for each window. The full feature map  $\mathbf{F}_j \in \mathbb{R}^{u_j \times H_j}$ , where  $H_j$  is total

<sup>5</sup>Details of these models can be found in the cited work and the appendix.

number of valid submatrices for the kernel, is:

$$\mathbf{F}_j = \sum_h (\mathbf{K}_j \text{vec}(\mathbf{M}^\sigma_{[h..k_j+h-1,*]}) + \mathbf{b}_j) \delta_h^\top. \quad (10)$$

Finally, an average pooling layer and a linear transform are applied to feature maps from different kernels:

$$\hat{\mathbf{f}} = [\text{mean}(\mathbf{F}_1) \dots \text{mean}(\mathbf{F}_J)]^\top, \quad (11)$$

$$\mathbf{e}^\sigma = \tanh(\mathbf{W}_{\text{pool}} \text{ReLU}(\hat{\mathbf{f}}) + \mathbf{b}_{\text{pool}}). \quad (12)$$

All  $\mathbf{K}$ s,  $\mathbf{b}$ s and  $\mathbf{W}$ s here are parameters of the projector.

## 5 Optimization

There are three different kinds of objectives used in the optimization of the full grounded induction model. The first loss is the marginal likelihood loss for the PCFG induction model described in Equation 1, which can be calculated with the Inside algorithm. The second loss is the syntactic-visual loss. Given the encoded image embedding  $\mathbf{e}^m$  and the projected syntactic-visual embedding  $\mathbf{e}^\sigma$  of a sentence  $\sigma$ , the syntactic-visual loss is the mean squared error of these two embeddings:

$$L(\mathbf{e}^m, \mathbf{e}^\sigma) = (\mathbf{e}^m - \mathbf{e}^\sigma)^\top (\mathbf{e}^m - \mathbf{e}^\sigma). \quad (13)$$

The third loss is the reconstruction loss of the image. Given the original image represented as a vector  $\mathbf{i}^m$  and the reconstructed image  $\hat{\mathbf{i}}^m$ , the reconstruction objective is the mean squared error of the corresponding pixel values of the two images:

$$L(m) = (\mathbf{i}^m - \hat{\mathbf{i}}^m)^\top (\mathbf{i}^m - \hat{\mathbf{i}}^m). \quad (14)$$

Models with different sets of input optimize the three losses differently for clean ablation. NoImagePCFG, which learns from text only, optimizes the negative marginal likelihood loss (the negative of Equation 1) using gradient descent. The model with pretrained image encoders, ImagePrePCFG, optimizes the negative marginal likelihood and the syntactic-visual loss (Equation 13) simultaneously. The full grounded grammar induction model ImagePCFG learns from text and images jointly by minimizing all three objectives: negative marginal likelihood, syntactic-visual loss and image reconstruction loss (Equation 14):

$$L(\sigma, m) = -P(\sigma) + L(\mathbf{e}^m, \mathbf{e}^\sigma) + L(m). \quad (15)$$

## 6 Experiment methods

Experiments described in this paper use the MSCOCO caption data set (Lin et al., 2015) and the Multi30k dataset (Elliott et al., 2016), which contains pairs of images and descriptions of images written by human annotators. Captions in the MSCOCO data set are in English, whereas captions in the Multi30k dataset are in English, German and French. Captions are automatically parsed (Kitaev and Klein, 2018) to generate a version of the reference set with constituency trees.<sup>6</sup> In addition to these datasets with captions generated by human annotators, we automatically translate the English captions into Chinese, Polish and Korean using Google Translate,<sup>7</sup> and parse the resulting translations into constituency trees, which are then used in experiments to probe the interactions between visual information and grammar induction.

Results from models proposed in this paper — NoImagePCFG, ImagePrePCFG and ImagePCFG — are compared with published results from Shi et al. (2019), which include PRPN (Shen et al., 2018), ON-LSTM (Shen et al., 2019) as well as the grounded VG-NSL models which uses either head final bias (VG-NSL<sub>H</sub>) or head final bias and Fasttext embeddings (VG-NSL<sub>H+F</sub>) as inductive biases from external sources. All of these models only induce unlabeled structures and have been evaluated with unlabeled F1 scores. We additionally report the labeled evaluation score Recall-Homogeneity (Rosenberg and Hirschberg, 2007; Jin and Schuler, 2020) for better comparison between the proposed models. All evaluation is done on Viterbi parse trees of the test set from 5 different runs. Details about hyper-parameters and results on development data sets can be found in the appendix. However, importantly, the tuned hyperparameters for the grammar induction model are the same across the three proposed models, which facilitates direct comparisons among these models to determine the effect of visual information on induction.

### 6.1 Standard set: no replication of effect for visual information

Both unlabeled and labeled evaluation results are shown in Table 1 with left- and right-branching baselines. First, trees induced by the PCFG induction models are more accurate than trees induced

<sup>6</sup>The multilingual parsing accuracy for all languages used in this work has been validated in Fried et al. (2019) and verified in Shi et al. (2019).

<sup>7</sup><https://translate.google.com/>.

with all other models, showing that the family of PCFG induction models is better at capturing syntactic regularities and provides a much stronger baseline for grammar induction. Second, using the NoImagePCFG model as a baseline, results from both the ImagePCFG model, where raw images are used as input, and the ImagePrePCFG model, where images encoded by pretrained image classifiers are used as input, do not show strong indication of benefits of visual information in induction. The baseline NoImagePCFG outperforms other models by significant margins on all languages in unlabeled evaluation. Compared to seemingly large gains between text-based models like PRPN and ON-LSTM<sup>8</sup> and the grounded models like VG-NSL<sub>H</sub> on French and German observed by Shi et al. (2019), the only positive gain between NoImagePCFG and ImagePCFG shown in Table 1 is the labeled evaluation on French where ImagePCFG outperforms NoImagePCFG by a small margin. Because the only difference between NoImagePCFG and ImagePCFG models is whether the visual information influences the syntactic word embeddings, the results indicate that on these languages, visual information does not seem to help induction. The gain seen in previous results may therefore be from external inductive biases. Finally, the ImagePrePCFG model performs at slightly lower accuracies than the ImagePCFG model consistently across different languages, datasets and evaluation metrics, showing that the information needed in grammar induction from images is not the same as information needed for image classification, and such information can be extracted from images without annotated image classification data.

### 6.2 Languages with wider distribution of high-frequency word types: positive effect

One potential advantage of using visual information in induction is to ground nouns and noun phrases. For example, if images like in Figure 1 are consistently presented to models with sentences describing *spaghetti*, the models may learn the categorize words and phrases which could be linked with objects in images as nominal units and then bootstrap other lexical categories. However, in the test languages above, a narrow set of very high fre-

<sup>8</sup>PCFG induction models where a grammar is induced generally perform better in parsing evaluation than sequence models where only syntactic structures are induced (Kim et al., 2019a; Jin et al., 2019).

Models	MSCOCO				Multi30k			
	English**		English**		German**		French**	
	F1	RH	F1	RH	F1	RH	F1	RH
Left-branching	23.3	-	22.6	-	34.7	-	19.0	-
Right-branching	21.4	-	11.3	-	12.1	-	11.0	-
PRPN	52.5 $\pm$ 2.6	-	30.8 $\pm$ 17.9	-	31.5 $\pm$ 8.9	-	27.5 $\pm$ 7.0	-
ON-LSTM	45.5 $\pm$ 3.3	-	38.7 $\pm$ 12.7	-	34.9 $\pm$ 12.3	-	27.7 $\pm$ 5.6	-
VG-NSL+H	53.3 $\pm$ 0.2	-	38.7 $\pm$ 0.2	-	38.3 $\pm$ 0.2	-	38.1 $\pm$ 0.6	-
VG-NSL+H+F	54.4 $\pm$ 0.4	-	-	-	-	-	-	-
NoImagePCFG	<b>60.0<math>\pm</math>8.2</b>	<b>47.6<math>\pm</math>10.0</b>	<b>59.4<math>\pm</math>7.7</b>	<b>51.6<math>\pm</math>8.5</b>	<b>48.1<math>\pm</math>5.2</b>	<b>53.7<math>\pm</math>5.2</b>	<b>44.3<math>\pm</math>5.1</b>	43.8 $\pm$ 5.2
ImagePrePCFG	55.6 $\pm$ 7.5	42.3 $\pm$ 7.3	47.0 $\pm$ 7.0	40.5 $\pm$ 7.2	46.2 $\pm$ 7.4	51.1 $\pm$ 8.0	42.6 $\pm$ 10.3	43.4 $\pm$ 10.8
ImagePCFG	55.1 $\pm$ 2.7	42.5 $\pm$ 1.5	48.2 $\pm$ 4.9	40.5 $\pm$ 5.0	47.0 $\pm$ 5.5	51.8 $\pm$ 8.4	43.6 $\pm$ 5.5	<b>44.5<math>\pm</math>6.3</b>

Table 1: Averages and standard deviations of labeled Recall-Homogeneity and unlabeled F1 scores of various unsupervised grammar inducers on the MSCOCO and Multi30k caption datasets. VG-NSL+H: VG-NSL system with head final bias. VG-NSL+H+F: VG-NSL system with head final bias and Fasttext word embeddings.(\*\* : the unlabeled performance difference between NoImagePCFG and ImagePCFG is significant  $p < 0.01$ .)

quency words such as determiners provide strong identifying information for nouns and noun phrases, which may greatly diminish the advantage contributed by visual information. In such cases, visual information may even be harmful, as models may attend to other information in images which is irrelevant to induction.

Korean, Polish and Chinese are chosen as representatives of languages with no definite articles, and in which statistical information provided by high frequency words is less reliable because there are more such word types. Table 2 shows the performance scores of the three proposed systems on these languages. Comparing to results in Table 1, the models with visual information in the input significantly outperform the baseline model, NoImagePCFG, on a majority of the additional test datasets. Figure 3 shows the correlation between the RH difference between the ImagePCFG model and the NoImagePCFG model on each language in an image dataset, and the distribution of high frequency words in that language, defined as the number of word types needed to account for 10% of the number of word tokens in the Universal Dependency (Nivre et al., 2016) corpus of a language.<sup>9</sup> The figure shows that the largest gain brought by visual information in induction is on Korean, where the number of high frequency word types is also highest. Results on Chinese and Polish

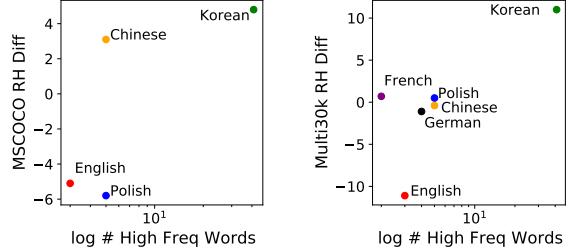


Figure 3: The correlation between number of word types needed to account for 10% of word tokens in a language ( $\log \# \text{High Freq Words}$ ) and the RH gain from NoImagePCFG to ImagePCFG on different languages on the two different image datasets.

also show a benefit for visual information, although the gain is much smaller and less consistent. It also shows that when there is a trend of positive correlation between the number of high frequency words and the gain brought by visual information, factors other than high frequency words are at play as well in determining the final induction outcome for each dataset in each language in the visually grounded setup, which are left for investigation in future work.

## 7 Analysis of advantages of visual information

We hypothesize three specific ways that visual information may help grammar induction. First, a strong correlation between words and objects in images can help identification and categorization

<sup>9</sup>Korean has 41, Chinese and Polish have 5, German has 4, English has 3 and French has 2.

Models on MSCOCO	Korean**		Polish**		Chinese**	
	F1	RH	F1	RH	F1	RH
NoImagePCFG	38.1 $\pm$ 8.5	22.3 $\pm$ 6.8	58.9 $\pm$ 3.7	47.1 $\pm$ 3.8	61.2 $\pm$ 3.5	48.5 $\pm$ 3.7
ImagePrePCFG	39.0 $\pm$ 4.1	23.5 $\pm$ 3.2	60.5 $\pm$ 1.8	49.8 $\pm$ 3.3	60.0 $\pm$ 4.6	47.2 $\pm$ 4.5
ImagePCFG	<b>45.0<math>\pm</math>2.2</b>	<b>27.1<math>\pm</math>2.6</b>	53.6 $\pm$ 8.3	41.3 $\pm$ 7.8	<b>64.9<math>\pm</math>6.6</b>	<b>51.2<math>\pm</math>8.6</b>

Models on Multi30k	Korean**		Polish		Chinese**	
	F1	RH	F1	RH	F1	RH
NoImagePCFG	30.7 $\pm$ 5.6	22.8 $\pm$ 3.1	49.6 $\pm$ 4.6	39.9 $\pm$ 5.1	<b>59.1<math>\pm</math>3.3</b>	<b>53.2<math>\pm</math>4.7</b>
ImagePrePCFG	27.1 $\pm$ 4.4	19.9 $\pm$ 3.4	48.4 $\pm$ 3.1	38.3 $\pm$ 2.9	57.9 $\pm$ 7.0	51.0 $\pm$ 7.7
ImagePCFG	<b>44.9<math>\pm</math>1.3</b>	<b>33.8<math>\pm</math>2.1</b>	<b>49.7<math>\pm</math>7.2</b>	<b>40.4<math>\pm</math>6.1</b>	58.5 $\pm$ 3.2	52.8 $\pm$ 4.6

Table 2: Averages and standard deviations of labeled Recall-Homogeneity and unlabeled F1 scores of various unsupervised grammar inducers on the MSCOCO and Multi30k caption datasets in the additional languages with high numbers of high-frequency word types. (\*\* : the unlabeled performance difference between NoImagePCFG and ImagePCFG is significant  $p < 0.01$ .)

of nouns and noun phrases, especially on languages where nouns and noun phrases are not readily identifiable by neighboring high frequency words. Second, visual information may provide bottom-up information for unknown word embeddings. Languages where neighboring words can reliably predict the grammatical category of an unknown word may build robust representations of unknown word embeddings, but the construction of the UNK embedding may also benefit from bottom-up information from images, especially when sentential context is not enough to build informative UNK embeddings. Finally, semantic information inside images may be helpful in solving syntactic ambiguities like prepositional phrase attachment in languages like English. Results from experiments described below with the ImagePCFG and NoImagePCFG models show evidence of all three ways.

### 7.1 Grounding of nouns and noun phrases

The ‘Noun bias’ hypothesis (Gentner, 1982) postulates that visual information in the induction process may impact how words are categorized grammatically, and nouns may receive an advantage because they correspond to objects in images. However, objects in images are often described with phrases, not single words. For example, captions like *a red car is parked on the street*, are common in both caption datasets, where the objects in the image may associate more strongly with modifier words like *red* than the head noun *car*.

Evaluations are carried out on the parsed sentences of all languages from two caption datasets

using a part-of-speech homogeneity metric (Rosenberg and Hirschberg, 2007) for measuring the part-of-speech accuracy, and an unlabeled NP recall score for measuring how many noun phrases in gold annotation are also found in the induced trees. Results in Figure 4 first show that the POS homogeneity scores from different models on the same induction dataset are extremely close to each other. Given that nouns are one of the categories with the most numerous tokens, the almost identical performance of POS homogeneity across different models indicates that the unsupervised clustering accuracy for nouns across different models is also very close, in contrast to substantial RH score differences on English and Korean.

However, NP recall scores show a pattern of performance ranking that resembles the ranking observed in Tables 1 and 2. For all datasets except for the Polish Multi30k dataset, when the RH score of ImagePCFG is higher than NoImagePCFG, the NP recall score for the ImagePCFG model is also higher. Significance testing with permutation sampling shows that all performance differences are significant ( $p < 0.01$ ).<sup>10</sup> High accuracy on noun phrases is crucial to high accuracy of other constituents such as prepositional phrases and verb phrases, which usually contain noun phrases, and eventually leads to high overall accuracy. This result suggests that the benefit contributed by visual information works at phrasal levels, most likely

<sup>10</sup>Significance testing is not done on POS homogeneity due to the possibility that the same induced POS label may mean different things in different induced grammars.

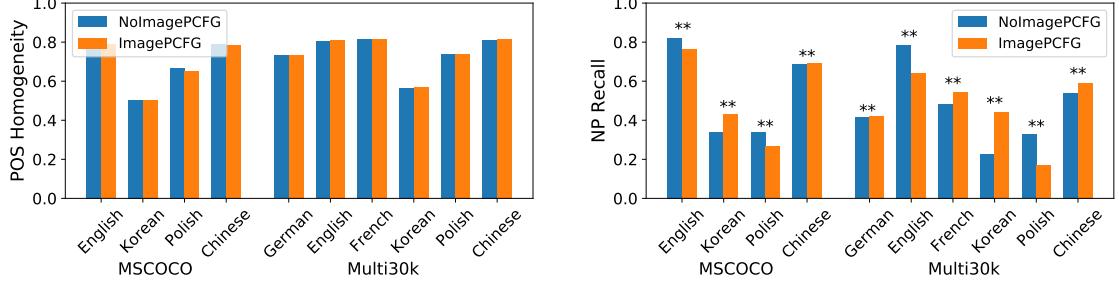


Figure 4: The POS Homogeneity and NP Recall scores for the ImagePCFG and NoImagePCFG models across the test languages (\*\* :  $p < 0.01$ ).

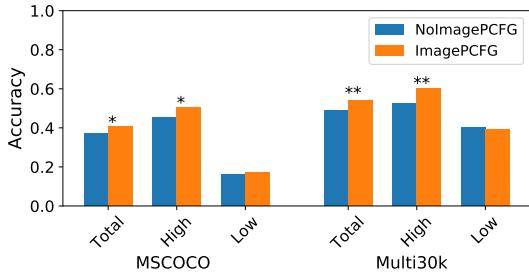


Figure 5: The average overall accuracy as well as accuracies for high and low attachment sentences in PP attachment evaluation for models with and without visual information (\*\* :  $p < 0.01$ , \* :  $p < 0.05$ ).

grounding phrases, not words, with objects in images.

## 7.2 Informativeness of the UNK embedding

The informativeness of unknown word embeddings is tested among the induction models across different languages. An UNK test set is created by randomly replacing one word in one sentence with an UNK symbol if the sentence has no unknown words present. Table 3 shows the labeled evaluation results on the multilingual datasets.<sup>11</sup> First, performance on the UNK test sets on all languages is lower than on the normal test sets, showing that replacing random words with UNK symbols does impact performance. The performance ranking of the models on a majority of the languages is consistent with the ranking on the normal test set. The ranking of the models on one dataset, the Chinese Multi30k, is reversed on the UNK test set, where the ImagePCFG models show significantly higher performance than the NoImagePCFG models (Chinese:  $p < 0.01$ , permutation test on unlabeled F1). This result indicates that the ImagePCFG model in which visual information is supplied during train-

<sup>11</sup>The unlabeled evaluation results can be found in the appendix.

ing may have built more informative embeddings for the unknown word symbols, helping the model to outperform the model without visual information on a majority of datasets where UNK symbols are frequent.

## 7.3 Prepositional phrase attachment

Finally, visual information may provide semantic information to resolve structural ambiguities. Word quintuples such as *(a) hotel caught fire during (a) storm* were extracted from English Wikipedia and the attachment locations were automatically labeled either as ‘n’ for low attachment, where the prepositional phrase adjoins the direct object, or ‘v’ for high attachment, where the prepositional phrase adjoins the main verb (Nakashole and Mitchell, 2015). 168 test items are selected by human annotators for evaluation, within which 119 are sentences with high attached PPs and 49 are with low attached PPs. For evaluation of PP attachment with induced trees, one test item is labeled correct when the induced tree puts the main verb and the direct object into one constituent and it is labeled as ‘v’. For example, if the induced tree has *caught fire* as a constituent, it counts as correct for the above example with high attachment. Low attachment trees must have a constituent with the direct object and the prepositional phrase. For example, for the sentence *(a) guide gives talks about animals*, the induced tree must have *talks about animals*. Average accuracies for all sentences as well as for sentences with high attachment or low attachment with induced grammars are shown in Figure 5. Results show that the models trained with visual information on both datasets show significantly higher performance on the PP attachment task in most of the categories, except for the low attachment category with Multi30k models where the performance from both models is not significantly different. This is in contrast to the

Models	MSCOCO				Multi30k					
	En	Ko	Pl	Zh	De	En	Fr	Ko	Pl	Zh
NoImagePCFG	<b>46.2</b>	21.7	<b>45.8</b>	46.0	<b>52.8</b>	<b>49.9</b>	42.2	22.8	38.9	<b>51.6</b>
ImagePCFG	41.2	<b>26.4</b>	40.2	<b>48.1</b>	51.3	39.9	<b>42.6</b>	<b>33.2</b>	<b>39.7</b>	<b>53.2</b>

Table 3: Average labeled Recall-Homogeneity of the NoImagePCFG and ImagePCFG models on the MSCOCO and Multi30k caption datasets with random words replaced by the UNK symbol. Standard deviations across the datasets are similar to what is reported in Table 1 and 2. Chinese Multi30k is the one on which the NoImagePCFG model outperforms the ImagePCFG model on the normal test set but not on the UNK test set.

higher performance of the NoImagePCFG models on unlabeled F1 and labeled RH than that of the ImagePCFG models on English from both caption datasets. Results indicate that induction models use visual information for weighting competing latent syntactic trees for a sentence, which is consistent with the third hypothesized advantage of visual information for induction. This also indicates that the reason that the overall parsing performance of ImagePCFG on English is lower than NoImagePCFG lies within other syntactic structures, which is left for future work.

## 8 Conclusion

This work proposed several novel neural network-based models of grammar induction which take into account visual information in induction. These models achieve state-of-the-art results on multilingual induction datasets without any help from linguistic knowledge or pretrained image encoders. Further analyses isolated three hypothesized benefits of visual information: it helps categorize noun phrases, represent unknown words and resolve syntactic ambiguities.

## Acknowledgments

The authors would like to thank the anonymous reviewers for their helpful comments. Computations for this project were partly run on the Ohio Supercomputer Center (1987). This work was supported by the Presidential Fellowship from the Ohio State University. The content of the information does not necessarily reflect the position or the policy of the Government, and no official endorsement should be inferred. This work was also supported by the National Science Foundation grant #1816891. All views expressed are those of the authors and do not necessarily reflect the views of the National Science Foundation.

## References

- Noam Chomsky. 1965. *Aspects of the Theory of Syntax*. MIT Press, Cambridge, MA.
- Gordon Christie, Ankit Laddha, Aishwarya Agrawal, Stanislaw Antol, Yash Goyal, Kevin Kochersberger, and Dhruv Batra. 2016. [Resolving language and vision ambiguities together: Joint segmentation & Prepositional attachment resolution in captioned scenes](#). In *EMNLP 2016 - Conference on Empirical Methods in Natural Language Processing, Proceedings*, pages 1493–1503.
- Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.
- Andrew Drozdov, Patrick Verga, Yi-Pei Chen, Mohit Iyyer, and Andrew McCallum. 2019. [Unsupervised labeled parsing with deep inside-outside recursive autoencoders](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1507–1512, Hong Kong, China. Association for Computational Linguistics.
- Desmond Elliott, Stella Frank, Khalil Sima'an, and Lucia Specia. 2016. [Multi30K: Multilingual English-German Image Descriptions](#). In *Proceedings of the 5th Workshop on Vision and Language*, pages 70–74, Berlin, Germany. Association for Computational Linguistics.
- Daniel Fried, Nikita Kitaev, and Dan Klein. 2019. [Cross-domain generalization of neural constituency parsers](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 323–330, Florence, Italy. Association for Computational Linguistics.
- Dedre Gentner. 1982. [Why nouns are learned before verbs: Linguistic relativity versus natural partitioning](#). *Language development: Vol. 2. Language, thought, and culture*, 2(1):301–334.
- Dedre Gentner. 2006. [Why Verbs Are Hard to Learn](#). In K. Hirsh-Pasek and R. Golinkoff, editors, *Action Meets Word: How Children Learn Verbs*, pages 544–564. Oxford University Press.

- Lila Gleitman. 1990. **The Structural Sources of Verb Meanings.** *Language Acquisition*, 1(1):3–55.
- Ajda Gokcen, Ethan Hill, and Michael White. 2018. **Madly ambiguous: A game for learning about structural ambiguity and why it’s hard for computers.** In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 51–55, New Orleans, Louisiana. Association for Computational Linguistics.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. **Deep residual learning for image recognition.** In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2016-Decem, pages 770–778.
- Felix Hill and Anna Korhonen. 2014. **Concreteness and subjectivity as dimensions of lexical meaning.** In *52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014 - Proceedings of the Conference*, volume 2, pages 725–731.
- Felix Hill, Roi Reichart, and Anna Korhonen. 2014. **Multi-Modal Models for Concrete and Abstract Concept Meaning.** *Transactions of the Association for Computational Linguistics*, 2:285–296.
- Lifeng Jin, Finale Doshi-Velez, Timothy Miller, William Schuler, and Lane Schwartz. 2018a. **Depth-bounding is effective: Improvements and evaluation of unsupervised PCFG induction.** In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2721–2731, Brussels, Belgium. Association for Computational Linguistics.
- Lifeng Jin, Finale Doshi-Velez, Timothy Miller, William Schuler, and Lane Schwartz. 2018b. **Unsupervised grammar induction with depth-bounded PCFG.** *Transactions of the Association for Computational Linguistics*, 6:211–224.
- Lifeng Jin, Finale Doshi-Velez, Timothy Miller, Lane Schwartz, and William Schuler. 2019. **Unsupervised learning of PCFGs with normalizing flow.** In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2442–2452, Florence, Italy. Association for Computational Linguistics.
- Lifeng Jin and William Schuler. 2019. **Variance of average surprisal: A better predictor for quality of grammar from unsupervised PCFG induction.** In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2453–2463, Florence, Italy. Association for Computational Linguistics.
- Lifeng Jin and William Schuler. 2020. **The Importance of Category Labels in Grammar Induction with Child-directed Utterances.** In *Proceedings of 16th International Conference on Parsing Technologies*, Seattle, USA. Association for Computational Linguistics.
- Mark Johnson, Thomas L. Griffiths, and Sharon Goldwater. 2007. **Bayesian Inference for PCFGs via Markov chain Monte Carlo.** *Proceedings of Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 139–146.
- Yoon Kim, Chris Dyer, and Alexander Rush. 2019a. **Compound probabilistic context-free grammars for grammar induction.** In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2369–2385, Florence, Italy. Association for Computational Linguistics.
- Yoon Kim, Alexander Rush, Lei Yu, Adhiguna Kunoro, Chris Dyer, and Gábor Melis. 2019b. **Unsupervised recurrent neural network grammars.** In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1105–1117, Minneapolis, Minnesota. Association for Computational Linguistics.
- Nikita Kitaev and Dan Klein. 2018. **Constituency parsing with a self-attentive encoder.** In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2676–2686, Melbourne, Australia. Association for Computational Linguistics.
- Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. 2015. **Microsoft COCO: Common Objects in Context.** In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3686–3693.
- Ndapandula Nakashole and Tom M Mitchell. 2015. **A knowledge-intensive model for prepositional phrase attachment.** In *ACL-IJCNLP 2015 - 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Proceedings of the Conference*, volume 1, pages 365–375.
- Joakim Nivre, Marie-Catherine De Marneffe, Filip Ginster, Yoav Goldberg, Jan Hajic, Christopher D Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. **Universal Dependencies v1: A Multilingual Treebank Collection.** In *Proceedings of Language Resources and Evaluation Conference*.
- Hiroshi Noji and Mark Johnson. 2016. **Using Left-corner Parsing to Encode Universal Structural Constraints in Grammar Induction.** In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 33–43.
- The Ohio Supercomputer Center.  
1987. Ohio Supercomputer Center.  
\url{http://osc.edu/ark:/19495/f5s1ph73}.

Steven Pinker and B MacWhinney. 1987. The bootstrapping problem in language acquisition. *Mechanisms of language acquisition*, pages 399–441.

Alec Radford, Luke Metz, and Soumith Chintala. 2016. Unsupervised representation learning with deep convolutional generative adversarial networks. In *Proceedings of the 4th International Conference on Learning Representations*. International Conference on Learning Representations, ICLR.

Andrew Rosenberg and Julia Hirschberg. 2007. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*.

Yoav Seginer. 2007. **Fast Unsupervised Incremental Parsing**. In *Proceedings of the Annual Meeting of the Association of Computational Linguistics*, pages 384–391.

Cory Shain, William Bryce, Lifeng Jin, Victoria Krakovna, Finale Doshi-Velez, Timothy Miller, William Schuler, and Lane Schwartz. 2016. **Memory-bounded left-corner unsupervised grammar induction on child-directed input**. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 964–975, Osaka, Japan. The COLING 2016 Organizing Committee.

Yikang Shen, Zhouhan Lin, Chin-Wei Huang, and Aaron Courville. 2018. **Neural Language Modeling by Jointly Learning Syntax and Lexicon**. In *ICLR*.

Yikang Shen, Shawn Tan, Alessandro Sordoni, and Aaron Courville. 2019. **Ordered Neurons: Integrating Tree Structures into Recurrent Neural Networks**. In *ICLR*.

Haoyue Shi, Jiayuan Mao, Kevin Gimpel, and Karen Livescu. 2019. **Visually grounded neural syntax acquisition**. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1842–1861, Florence, Italy. Association for Computational Linguistics.

Michael Tomasello. 2003. *Constructing a language: A usage-based theory of language acquisition*. Harvard University Press, Cambridge, MA, US.

Sandra Waxman, Xiaolan Fu, Sudha Arunachalam, Erin Leddon, Kathleen Geraghty, Hyun-Joo Song, Child Dev, and Perspect Author. 2013. **Are Nouns Learned Before Verbs? Infants Provide Insight into a Longstanding Debate NIH Public Access Author Manuscript**. *Child Dev Perspect*, 7(3).

## A Details of datasets

The MSCOCO caption dataset used in Shi et al. (2019) contains 413,915 sentences in the training set, and 5000 sentences in the development and test sets respectively.<sup>12</sup> Every image is accompanied by 5 captions, and there are 82,783 images in total in the training set. The image embeddings of size 2048 used in Shi et al. (2019) are encoded by an image classifier with ResNet128 architecture trained with on the ImageNet classification task (Deng et al., 2009).

The Multi30k caption dataset contains 29,000 sentences in the training set, and 1,014 sentences in the development and 1,000 in the test set in four different languages, all of which except Czech are used in this work thanks to the availability of high accuracy constituency parsers in these languages.<sup>13</sup> There are as many images as there are captions in the training set. The image embeddings of size 2048 provided with the dataset are encoded by an image classifier with ResNet50 architecture also trained with on the ImageNet classification task.

For data preprocessing, following Shi et al. (2019), the size of the vocabulary is limited to 10,000 for all languages and datasets. All raw images are resized to  $3 \times 64 \times 64$  and normalized with means [0.485, 0.456, 0.406] and standard deviations [0.229, 0.224, 0.225], calculated from images in ImageNet.

## B Hyperparameters

The hyperparameters used in all proposed models are tuned with the MSCOCO English development set. For the grammar induction model, the size of word and syntactic category embeddings, as well as the size of hidden intermediary representations is 64. The size of the image embedding in the ImagePCFG system is also 64. All out-of-vocabulary words are replaced by the UNK symbol. Sentences with more than 40 words in the training set are trimmed down to 40 words. For the projector model, five different convolutional kernels, from (1,64) to (5,64), are used with 128 output channels. The trainable image encoder employs a

<sup>12</sup>The data set can be found at <https://github.com/ExplorerFreda/VGNSL> along with image embeddings encoded by pretrained image encoders.

<sup>13</sup>The data set can be found at <https://github.com/multi30k/dataset> along with image embeddings encoded by pretrained image encoders.

ResNet18 architecture,<sup>14</sup> and the decoder employs the decoder architecture in the DCGAN model.<sup>15</sup>

A batch size of 2 is used in training. Adam is used as the optimizer, with the initial learning rate at  $5 \times 10^{-4}$ . The loss on the validation set is checked every 20000 batches, and training is stopped when the validation loss has not been lowered for 10 checkpoints. The model with the lowest validation loss is used as the candidate model for test evaluation, where best parses are generated with the Viterbi algorithm on an inside chart.

## C Development

Table 4 and 5 report unlabeled F1 and labeled RH results on the development sets in the multilingual caption datasets. Results show that development and test results are very similar, indicating that the general characteristics of the two sets are very close.

<sup>14</sup>[https://pytorch.org/docs/stable/\\_modules/torchvision/models/resnet.html#resnet18](https://pytorch.org/docs/stable/_modules/torchvision/models/resnet.html#resnet18)

<sup>15</sup><https://github.com/pytorch/examples/blob/master/dcgan/main.py>

Models	English		Korean		Polish		Chinese	
	F1	RH	F1	RH	F1	RH	F1	RH
NoImagePCFG	60.3 $\pm$ 8.2	46.4 $\pm$ 11.0	38.6 $\pm$ 8.7	22.6 $\pm$ 6.9	59.5 $\pm$ 3.8	47.5 $\pm$ 3.9		
ImagePrePCFG	55.7 $\pm$ 7.5	39.6 $\pm$ 5.4	39.5 $\pm$ 4.2	24.1 $\pm$ 3.4	61.2 $\pm$ 1.6	50.1 $\pm$ 3.3		
ImagePCFG	55.4 $\pm$ 2.7	43.2 $\pm$ 1.8	45.1 $\pm$ 2.3	27.5 $\pm$ 2.6	54.3 $\pm$ 8.3	41.6 $\pm$ 7.9		

Table 4: Averages and standard deviations of labeled Recall-Homogeneity and unlabeled F1 scores of various unsupervised grammar inducers on the MSCOCO caption development datasets.

Models	German		English		French	
	F1	RH	F1	RH	F1	RH
NoImagePCFG	47.2 $\pm$ 5.7	53.6 $\pm$ 5.7	59.1 $\pm$ 8.1	52.2 $\pm$ 8.5	43.8 $\pm$ 4.9	43.2 $\pm$ 5.2
ImagePrePCFG	44.8 $\pm$ 7.9	50.0 $\pm$ 8.3	46.7 $\pm$ 7.3	40.7 $\pm$ 7.5	42.3 $\pm$ 10.3	42.8 $\pm$ 10.5
ImagePCFG	45.6 $\pm$ 5.2	50.6 $\pm$ 8.5	47.7 $\pm$ 5.4	40.9 $\pm$ 5.2	43.1 $\pm$ 5.1	43.9 $\pm$ 5.5

Models	Korean		Polish		Chinese	
	F1	RH	F1	RH	F1	RH
NoImagePCFG	30.6 $\pm$ 5.7	22.2 $\pm$ 3.0	49.4 $\pm$ 4.9	40.0 $\pm$ 5.3	59.7 $\pm$ 3.3	53.6 $\pm$ 4.7
ImagePrePCFG	27.0 $\pm$ 4.8	19.2 $\pm$ 3.6	48.5 $\pm$ 3.1	38.5 $\pm$ 3.1	55.5 $\pm$ 9.3	48.3 $\pm$ 10.4
ImagePCFG	45.1 $\pm$ 1.1	33.4 $\pm$ 1.9	49.5 $\pm$ 7.6	40.8 $\pm$ 6.3	58.3 $\pm$ 3.2	52.1 $\pm$ 4.3

Table 5: Averages and standard deviations of labeled Recall-Homogeneity and unlabeled F1 scores of various unsupervised grammar inducers on the Multi30k caption development datasets.

# Heads-up! Unsupervised Constituency Parsing via Self-Attention Heads

Bowen Li<sup>†</sup> Taeuk Kim<sup>‡</sup> Reinald Kim Amplayo<sup>†</sup> Frank Keller<sup>†</sup>

<sup>†</sup>ILCC, School of Informatics, University of Edinburgh, UK

<sup>‡</sup>Dept. of Computer Science and Engineering, Seoul National University, Korea

{bowen.li, reinald.kim}@ed.ac.uk

taeuk@europa.snu.ac.kr keller@inf.ed.ac.uk

## Abstract

Transformer-based pre-trained language models (PLMs) have dramatically improved the state of the art in NLP across many tasks. This has led to substantial interest in analyzing the syntactic knowledge PLMs learn. Previous approaches to this question have been limited, mostly using test suites or probes. Here, we propose a novel fully unsupervised parsing approach that extracts constituency trees from PLM attention heads. We rank transformer attention heads based on their inherent properties, and create an ensemble of high-ranking heads to produce the final tree. Our method is adaptable to low-resource languages, as it does not rely on development sets, which can be expensive to annotate. Our experiments show that the proposed method often outperform existing approaches if there is no development set present. Our unsupervised parser can also be used as a tool to analyze the grammars PLMs learn implicitly. For this, we use the parse trees induced by our method to train a neural PCFG and compare it to a grammar derived from a human-annotated treebank.

## 1 Introduction

Pre-trained language models (PLMs), particularly BERT (Devlin et al., 2019) and others (Yang et al., 2019; Liu et al., 2019b; Radford et al., 2019) based on the transformer architecture (Vaswani et al., 2017), have dramatically improved the state of the art in NLP. Such models make it possible to train a large, generic language model on vast unannotated datasets, and then fine-tune it for a specific task using a small amount of annotated data. The success of PLMs has led to a large literature investigating the linguistic knowledge that PLMs learn implicitly during pre-training (Liu et al., 2019a; Clark et al., 2019; Kovaleva et al., 2019; Pimentel et al., 2020), sometimes referred to as BERTology (Rogers et al., 2020).

BERTology has been particularly concerned with the question whether BERT-type models learn syntactic structure. Typical approaches include test suites of sentences that instantiate specific syntactic structures (Goldberg, 2019), general probes (also known as diagnostic classifiers, Belinkov and Glass 2019) or structural probes (Hewitt and Manning, 2019). All of these approaches are limited: the first one requires the laborious compilation of language- and construction-specific suites of sentences; the second one sometimes fails to adequately reflect differences in representations (Zhang and Bowman, 2018; Hewitt and Liang, 2019; Voita and Titov, 2020); the third one involves designing a novel extraction model that is not applicable to tasks other than probing (Maudslay et al., 2020).

It is therefore natural to use a parsing task to test whether the representations learned by PLMs contain usable syntactic information. This enables us to test syntactic structure in general, rather than specific constructions, and doesn't require a specialized probe. In this paper, we will therefore use PLM attention heads to construct an *unsupervised constituency parser*. Previously, related approaches have been proposed under the heading of *zero-shot* constituency parsing (Kim et al., 2020a,b).<sup>1</sup> However, this prior work crucially relies on an annotated development set in order to identify transformer heads that are sensitive to syntactic structure. Existing approaches therefore are not truly unsupervised. For most low resource languages, no such annotated data is available, and often not even an annotation scheme exists. Thus, assuming a development set is not a realistic experimental setup (Kann et al., 2019). If a suitable development set is available, Shi et al. (2020) shows that an existing supervised parser trained on a few-shot setting can outperform all the unsupervised parsing

<sup>1</sup>Like Kim et al. (2020b), we use *zero-shot* to refer to the transfer from language modeling to constituency parsing.

methods by a significant margin. It strongly challenges tuning on an annotated development set for unsupervised parsing.

In this paper, we propose a novel approach to build a PLM-based unsupervised parser that does not require a development set: we rank transformer heads based on their inherent properties, such as how likely tokens are to be grouped in a hierarchical structure. We then ensemble the top- $K$  heads to produce constituency trees.

We evaluate our approach and previous zero-shot approaches on the English Penn Treebank (PTB) and eight other languages on the SPMRL dataset. On the one hand, if the development set is absent, our approach largely outperforms previous zero-shot approaches on the English PTB. On the other hand, if previous zero-shot approaches are equipped with the development set, our approach can still match the parsing performance of these approaches using the single best head or layer-wise ensembling. For the multilingual experiment, we take advantage of the top- $K$  heads selected in English and directly parse other languages using our approach. Surprisingly, on five out of nine languages, this *crosslingual* unsupervised parser matches previous approaches that rely on a development set in each target language with the single best head or layer-wise ensembling. However, our fully unsupervised method lags behind the previous state-of-the-art zero-shot parser if a top- $K$  ensemble is used.

Furthermore, our approach can be used as a tool to analyze the capability of PLMs in learning syntactic knowledge. As no human annotation is required, our approach has the potential to reveal the grammar PLMs learn implicitly. Here, we use the tree structures generated by our parser to train a neural PCFG. We evaluate the learned grammar against the English PTB on internal tags and production rules both qualitatively and quantitatively.

## 2 Related Work

Recently, neural models have renewed interest in grammar induction. Earlier work (Choi et al., 2018; Williams et al., 2018) attempted to induce grammar by optimizing a sentence classification objective, while follow-up work (Htut et al., 2018; Shen et al., 2018a, 2019) showed that a language modeling objective performs better. Latest work employed autoencoders or probabilistic grammars (Drozdov et al., 2019; Kim et al., 2019a,b; Zhu et al., 2020).

A new line of work is zero-shot constituency parsing, whose goal is to automatically extract trees from PLMs in a parameter-free fashion. The top-down zero-shot parser (Kim et al., 2020a) utilizes the concept of *syntactic distance* (Shen et al., 2018b), where trees are induced by an algorithm that recursively splits a sequence of words in a top-down manner. However, this approach suffers from its greedy search mode, failing to take into account all possible subtrees. The chart-based zero-shot parser (Kim et al., 2020b) applies chart parsing to address this problem. Wu et al. (2020) introduced a parameter-free probing technique to analyze PLMs via perturbed masking.

There is also prior work on extracting constituency trees from self-attention mechanisms of transformers. Mareček and Rosa (2018) proposed heuristic approaches to convert attention weights to trees. Mareček and Rosa (2019) introduced a chart-based tree extraction method in transformer-based neural machine translation encoders and provide a quantitative study.

## 3 Zero-shot Constituency Parsing via PLMs

In this section, we briefly review the chart-based zero-shot parser and then introduce our ranking-based zero-shot parser.

### 3.1 Chart-based Zero-shot Parsing

In chart-based zero-shot parsing, a real-valued score  $s_{tree}(\mathbf{t})$  is assigned for each tree candidate  $\mathbf{t}$ , which decomposes as:

$$s_{tree}(\mathbf{t}) = \sum_{(i,j) \in \mathbf{t}} s_{span}(i, j),$$

where  $s_{span}(i, j)$  is the score (or cost) for a constituent that is located between positions  $i$  and  $j$  ( $1 \leq i \leq j \leq n$ , where  $n$  is the length of the sentence). Specifically, for a span of length 1,  $s_{span}(i, j)$  is defined as 0 when  $i = j$ . For a span longer than 1, the following recursion applies:

$$s_{span}(i, j) = s_{comp}(i, j) + \min_{i \leq k < j} s_{split}(i, k, j) \quad (1)$$

$$s_{split}(i, k, j) = s_{span}(i, k) + s_{span}(k + 1, j), \quad (2)$$

where  $s_{comp}(\cdot, \cdot)$  measures the validity or compositionality of the span  $(i, j)$  itself, while  $s_{split}(i, k, j)$  indicates how plausible it is to split the span  $(i, j)$  at position  $k$ . Two alternatives have been developed in Kim et al. (2020b) for  $s_{comp}(\cdot, \cdot)$ : the pair

score function  $s_p(\cdot, \cdot)$  and the characteristic score function  $s_c(\cdot, \cdot)$ .

The pair score function  $s_p(\cdot, \cdot)$  computes the average pair-wise distance in a given span:

$$s_p(i, j) = \frac{1}{\binom{j-i+1}{2}} \sum_{(w_x, w_y) \in \text{pair}(i, j)} f(g(w_x), g(w_y)), \quad (3)$$

where  $\text{pair}(i, j)$  returns a set consisting of all combinations of two words (e.g.,  $w_x, w_y$ ) inside the span  $(i, j)$ .

Functions  $f(\cdot, \cdot)$  and  $g(\cdot)$  are the distance measure function and the representation extractor function, respectively. For  $g$ , given  $l$  as the number of layers in a PLM,  $g$  is actually a set of functions  $g = \{g_{(u,v)}^d | u = 1, \dots, l, v = 1, \dots, a\}$ , each of which outputs the attention distribution of the  $v^{th}$  attention head on the  $u^{th}$  layer of the PLM.<sup>2</sup> In case of the function  $f$ , there are also two options, Jensen-Shannon (JSD) and Hellinger (HEL) distance. Thus,  $f = \{\text{JSD}, \text{HEL}\}$ .

The characteristic score function  $s_c(\cdot, \cdot)$  measures the distance between each word in the constituent and a predefined characteristic value  $c$  (e.g., the center of the constituent):

$$s_c(i, j) = \frac{1}{j-i+1} \sum_{i \leq x \leq j} f(g(w_x), c), \quad (4)$$

where  $c = \frac{1}{j-i+1} \sum_{i \leq y \leq j} g(w_y)$ .

Since  $s_{comp}(\cdot, \cdot)$  is well defined, it is straightforward to compute every possible case of  $s_{span}(i, j)$  using the CKY algorithm (Cocke, 1969; Kasami, 1966; Younger, 1967). Finally, the parser outputs  $\hat{t}$ , the tree that requires the lowest score (cost) to build, as a prediction for the parse tree of the input sentence:  $\hat{t} = \arg \min_t s_{tree}(t)$ .

For attention heads ensembling, both a layer-wise ensemble and a top- $K$  ensemble are considered. The first one averages all attention heads from a specific layer, while the second one averages the top- $K$  heads from across different layers. At test time, separate trees produced by different heads are merged to one final tree via *syntactic distance*.<sup>3</sup> The chart-based zero-shot parser achieves the state of the art in zero-shot constituency parsing.

<sup>2</sup>The hidden representations of the given words can also serve as an alternative for  $g$ . But Kim et al. (2020a) show that the attention distributions provide more syntactic clues under the zero-shot setting.

<sup>3</sup>Details can be found in Kim et al. (2020b). For the ensemble parsing, marrying chart-based parser and top-down parser yields better results than averaging the attention distributions.

### 3.2 Ranking-based Zero-shot Parsing

The chart-based zero-shot parser relies on the existing development set of a treebank (e.g., the English PTB) to select the best configuration, i.e., the combination of  $\{g | g_{(u,v)}^d, u = 1, \dots, l, v = 1, \dots, a\}$ ,  $\{f | \text{JSD, HEL}\}$ ,  $\{s_{comp} | s_p, s_c\}$ , and heads ensemble that achieves the best parsing accuracy. Such a development set always contains hundreds of sentences, hence considerable annotation effort is still required. From the perspective of unsupervised parsing, such results arguably are not fully unsupervised.<sup>4</sup> Another argument against using a development set is that the linguistic assumptions inherent in the expert annotation required to create the development set potentially restrict our exploration of how PLMs model the constituency structures. It could be that the PLM learns valid constituency structures, which however do not match the annotation guidelines that were used to create the development set.

Here, we take a radical departure from the previous work in order to extract constituency trees from PLMs in a fully unsupervised manner. We propose a two-step procedure for unsupervised parsing: (1) identify syntax-related attention heads directly from PLMs without relying on a development set of a treebank; (2) ensemble the selected top- $K$  heads to produce the constituency trees.

For identification of the syntax-related attention heads, we rank all heads by scoring them with a chart-based ranker. We borrow the idea of the chart-based zero-shot parser to build our ranker. Given an input sentence and a specific choice of  $f$  and  $s_{comp}$ , each attention head  $g_{(u,v)}^d$  in the PLM yields one unique attention distribution. Using the chart-based zero-shot parser in Section 3.1, we can obtain the score of the best constituency tree as:<sup>5</sup>

$$s_{parsing}(u, v) = s_{tree}(\hat{t}) = \sum_{(i,j) \in \hat{t}} s_{span}(i, j), \quad (5)$$

where  $\hat{t} = \arg \min_t s_{tree}(t)$ . It is obvious

<sup>4</sup>Some previous work (Shen et al., 2018a, 2019; Drozdov et al., 2019; Kim et al., 2019a) also use a development set to tune hyperparameters or early-stop training.

<sup>5</sup>Our ranking method works approximately as a maximum a posteriori probability (MAP) estimate, since we only consider the best tree the attention head generates. In unsupervised parsing, marginalization is a standard method for model development. We have tried to apply marginalization to our ranking algorithm where all possible trees are considered and the sum score is calculated (using the logsumexp trick) for ranking. But marginalization does not work well for attention distributions, where an “attending broadly” head with higher entropy is more favorable than a syntax-related head with lower entropy. So we only consider the score of the best tree.

that all combinations of  $\{f \mid \text{JSD}, \text{HEL}\}$  and  $\{s_{\text{comp}} \mid s_p, s_c\}$  will produce multiple scores for a given head. Here we average the scores of all such combinations to get one single score. Then we rank all attention heads and select the syntax-related heads for parsing. However, directly applying the chart-based zero-shot parser in Section 3.1 for ranking delivers a trivial, ill-posed solution. The recursion in Eq. (2) only encourages the intra-similarity inside the span. Intuitively, one attention head that produces the *same* attention distribution for each token (e.g., a uniform attention distribution or one that forces every token to attend to one specific token) will get the lowest score (cost) and the highest ranking.<sup>6</sup>

To address this issue, we first introduce inter-similarity into the recursion in Eq. (2) and get the following:

$$s_{\text{split}}(i, k, j) = s_{\text{span}}(i, k) + s_{\text{span}}(k+1, j) - s_{\text{cross}}(i, k, j), \quad (6)$$

where the cross score  $s_{\text{cross}}(i, k, j)$  is the similarity between two subspans  $(i, k)$  and  $(k+1, j)$ . However, this formulation forces the algorithm to go to the other extreme: one attention head that produces a totally *different* distribution for each token (e.g., force each token to attend to itself or the previous/next token) will get the highest ranking. To balance the inter- and intra-similarity and avoid having to introduce a tunable coefficient, we simply add a length-based weighting term to Eq. (1) and get:

$$s_{\text{span}}(i, j) = \frac{j-i+1}{n} (s_{\text{comp}}(i, j) + \min_{i \leq k < j} s_{\text{split}}(i, k, j)), \quad (7)$$

where  $j-i+1$  is the length of the span  $(i, j)$ . The length ratio functions as a regulator to assign larger weights to longer spans. This is motivated by the fact that longer constituents should contribute more to the scoring of the parse tree, since the inter-similarity always has strong effects on shorter spans. In this way, the inter- and intra-similarity can be balanced.

With respect to the choice for  $s_{\text{cross}}(i, k, j)$ , we follow the idea of  $s_p$  and  $s_c$  in Eq. (3) and (4)

<sup>6</sup>Such cases do exist in PLMs. Clark et al. (2019) shows that BERT exhibits clear surface-level attention patterns. Some of these patterns will deliver ill-posed solutions in ranking: attend broadly, attend to a special tokens (e.g., [SEP]), attend to punctuation (e.g., period). One can also observe these patterns using the visualization tool provided by Vig (2019).

and propose the pair score function  $s_{px}$  and the characteristic score function  $s_{cx}$ <sup>7</sup> for cross score computation.  $s_{px}$  is defined as:

$$s_{px}(i, j) = \frac{1}{(k-i+1)(j-k)} \sum_{(w_x, w_y) \in \text{prod}(i, k, j)} f(g(w_x), g(w_y)),$$

where  $\text{prod}(i, k, j)$  returns a set of the product of words from the two subspans  $(i, k)$  and  $(k+1, j)$ . And  $s_{cx}$  is defined as:

$$s_{cx}(i, j) = f(\mathbf{c}_{i,k}, \mathbf{c}_{k+1,j}),$$

where  $\mathbf{c}_{i,k} = \frac{1}{k-i+1} \sum_{i \leq x \leq k} g(w_x)$ ,  $\mathbf{c}_{k+1,j} = \frac{1}{j-k} \sum_{k+1 \leq y \leq j} g(w_y)$ .

We average all the combinations of  $\{f \mid \text{JSD}, \text{HEL}\}$ ,  $\{s_{\text{comp}} \mid s_p, s_c\}$  and  $\{s_{\text{cross}} \mid s_{px}, s_{cx}\}$  to rank all the attention heads and select the top- $K$  heads. After the ranking step, we perform constituency parsing by ensembling the selected heads. We simply employ the ensemble method in Section 3.1 and average all the combinations of  $\{f \mid \text{JSD}, \text{HEL}\}$  and  $\{s_{\text{comp}} \mid s_p, s_c\}$  to get a single predicted parse tree for a given sentence.

### 3.3 How to select $K$

For ensemble parsing, Kim et al. (2020b) proposed three settings: the best head, layerwise ensemble, and top- $K$  ensemble. To prevent introducing a tunable hyperparameter, we propose to select a value for  $K$  dynamically based on a property of the ranking score in Eq. (5).

Since we use a similarity-based distance, the lower the ranking score, the higher the ranking. Assuming that scores are computed for all attention heads, we can sort the scores in ascending order. Intuitively, given the order, we would like to choose the  $k$  for which ranking score increases the most, which means syntactic relatedness drops the most. Suppose  $s_{\text{parsing}}(k)$  is the ranking score where  $k$  is the head index in the ascending order, then this is equivalent to finding the  $k$  with the greatest gradient on the curve of the score. We first estimate the gradient of  $s_{\text{parsing}}(k)$  and then find the  $k$  with the greatest gradient. Finally,  $K$  is computed as:

$$K = \arg \max_k \sum_{\substack{k-\delta \leq j \leq k+\delta \\ j \neq k}} \frac{s_{\text{parsing}}(k+j) - s_{\text{parsing}}(k)}{j},$$

<sup>7</sup>Subscripts in the naming of functions in this paper:  $p$  – pair score,  $c$  – characteristic score,  $x$  – cross score.

where we smooth the gradient by considering  $\delta$  steps. Here, we set  $\delta = 3$ .

In practice, we find that the greatest gradient always happens in the head or the tail of the curve. For the robustness, we select the  $K$  from the middle range of the score function curve, i.e., starting from 30 and ending with 75% of all heads.<sup>8</sup> We also provide a *lazy* option for  $K$  selection, which simply assume a fixed value of 30 for the top- $K$  ensemble.

## 4 Grammar Learning

We are also interested in exploring to what extent the syntactic knowledge acquired by PLMs resembles human-annotated constituency grammars. For this exploration, we infer a constituency grammar, in the form of probabilistic production rules, from the trees induced from PLMs. This grammar can then be analyzed further, and compared to human-derived grammars. Thanks to the recent progress in neural parameterization, neural PCFGs have been successfully applied to unsupervised constituency parsing (Kim et al., 2019a). We harness this model<sup>9</sup> to learn probabilistic constituency grammars from PLMs by maximizing the joint likelihood of sentences and parse trees induced from PLMs. In the following, we first briefly review the neural PCFG and then introduce our training algorithm.

### 4.1 Neural PCFGs

A probabilistic context-free grammar (PCFG) consists of a 5-tuple grammar  $\mathcal{G} = (S, \mathcal{N}, \mathcal{P}, \Sigma, \mathcal{R})$  and rule probabilities  $\pi = \{\pi_r\}_{r \in \mathcal{R}}$ , where  $S$  is the start symbol,  $\mathcal{N}$  is a finite set of nonterminals,  $\mathcal{P}$  is a finite set of preterminals,  $\Sigma$  is a finite set of terminal symbols, and  $\mathcal{R}$  is a finite set of rules associated with probabilities  $\pi$ . The rules are of the form:

$$\begin{aligned} S &\rightarrow A, & A &\in \mathcal{N} \\ A &\rightarrow BC, & A &\in \mathcal{N}, \quad B, C \in \mathcal{N} \cup \mathcal{P} \\ T &\rightarrow w, & T &\in \mathcal{P}, w \in \Sigma. \end{aligned}$$

---

<sup>8</sup>Although our ranking algorithm can filter out *noisy* heads, by observing the attention heatmaps, we find that noisy heads sometimes still rank high. We do not do any post-processing to further filter out the noisy heads, so we empirically search  $k$  starting at 30.

<sup>9</sup>A more advanced version of the neural PCFG, the compound PCFG, has also been developed in Kim et al. (2019a). In this model variant, a compound probability distribution is built upon the parameters of a neural PCFG. In preliminary experiments, we found the compound PCFG learns similar grammars as the neural PCFG. So we only use the more lightweight neural PCFG in this work.

Assuming  $\mathcal{T}_{\mathcal{G}}$  is the set of all possible parse trees of  $\mathcal{G}$ , the probability of a parse tree  $t \in \mathcal{T}_{\mathcal{G}}$  is defined as  $p(t) = \prod_{r \in t_{\mathcal{R}}} \pi_r$ , where  $t_{\mathcal{R}}$  is the set of rules used in the derivation of  $t$ . A PCFG also defines the probability of a given sentence  $x$  (string of terminals  $x \in \Sigma^*$ ) via  $p(x) = \sum_{t \in \mathcal{T}_{\mathcal{G}}(x)} p(t)$ , where  $\mathcal{T}_{\mathcal{G}}(x) = \{t | \text{yield}(t) = x\}$ , i.e., the set of trees  $t$  such that  $t$ 's leaves are  $x$ .

The traditional way to parameterize a PCFG is to assign a scalar to each rule  $\pi_r$  under the constraint that valid probability distributions must be formed. For unsupervised parsing, however, this parameterization has been shown to be unable to learn meaningful grammars from natural language data (Carroll and Charniak, 1992). Distributed representations, the core concept of the modern deep learning, have been introduced to address this issue (Kim et al., 2019a). Specifically, embeddings are associated with symbols and rules are modeled based on such distributed and shared representations.

In the neural PCFG, the log marginal likelihood:

$$\log p_{\theta}(x) = \log \sum_{t \in \mathcal{T}_{\mathcal{G}}(x)} p_{\theta}(t)$$

can be computed by summing out the latent parse trees using the inside algorithm (Baker, 1979), which is differentiable and amenable to gradient based optimization. We refer readers to the original paper of Kim et al. (2019a) for details on the model architecture and training scheme.

### 4.2 Learning Grammars from Induced Trees

Given the trees induced from PLMs (described in Section 3.2), we use neural PCFGs to learn constituency grammars. In contrast to unsupervised parsing, where neural PCFGs are trained solely on raw natural language data, we train them on the sentences and the corresponding tree structures induced from PLMs. Note that this differs from a fully supervised parsing setting, where both tree structures and internal constituency tags (nonterminals and preterminals) are provided in the treebank. In our case, the trees induced from PLMs have no internal annotations.

For the neural PCFG training, the joint likelihood is given by:

$$\log p(x, \hat{t}) = \sum_{r \in \hat{t}_{\mathcal{R}}} \log \pi_r,$$

where  $\hat{t}$  is the induced tree and  $\hat{t}_{\mathcal{R}}$  is the set of rules applied in the derivation of  $\hat{t}$ . Although tree struc-

tures are given during training, marginalization is still involved: all internal tags will be marginalized to compute the joint likelihood. Therefore, the grammars learned by our method are anonymized: nonterminals and preterminals will be annotated as  $NT\text{-}id$  and  $T\text{-}id$ , respectively, where  $id$  is an arbitrary ID number.

## 5 Experiments

We conduct experiments to evaluate the unsupervised parsing performance of our ranking-based zero-shot parser on English and eight other languages (Basque, French, German, Hebrew, Hungarian, Korean, Polish, Swedish). For the grammars learned from the induced parse trees, we perform qualitative and quantitative analysis on how the learned grammars resemble the human-crafted grammar of the English PTB.

### 5.1 General Setup

We prepare the PTB (Marcus et al., 1993) for English and the SPMRL dataset (Seddah et al., 2013) for eight other languages. We adopt the standard split of each dataset to divide it into development and test sets. For preprocessing, we follow the setting in Kim et al. (2019a,b).

We run our ranking algorithm on the development set to select the syntax-related heads and the ensemble parsing algorithm on the test set. We only use the raw sentences in the development set, without any syntactic annotations. We average all configurations both for ranking ( $f$ ,  $s_{comp}$  and  $s_{cross}$ ) and parsing ( $f$  and  $s_{comp}$ ); hence we do not tune any hyperparameters for our algorithm. For  $K$  selection, we experiment with fixed top- $K$  (i.e., top-30) and dynamically searching the best  $K$  described in Section 3.3, dubbed dynamic  $K$ . We report the unlabeled sentence-level  $F_1$  score to evaluate the extent to which the induced trees resemble the corresponding gold standard trees.

For neural PCFG training, we modify some details but keep most of the model configurations of Kim et al. (2019a); we refer readers to the original paper for more information. We train the models on longer sentences for more epochs. Specifically, we train on sentences of length up to 30 in the first epoch, and increase this length limit by five until the length reaches 80. We train for 30 epochs and use a learning rate scheduler.

Model Configuration	Top-down		Chart-based		Our ranking-based		
	Single /Layer <sup>†</sup>	Single /Layer <sup>†</sup>	Top - $K$	Top - $K$ <sup>‡</sup>	Top - $K$	Dynamic $K$	Full heads
	w/ dev trees		w/o dev trees				
BERT-base-cased	32.6	37.5	<b>42.7</b>	29.3	34.8	<b>37.1</b>	35.8
BERT-large-cased	36.7	41.5	<b>44.6</b>	21.5	36.1	<b>38.7</b>	33.2
XLNet-base-cased	39.0	40.5	<b>46.4</b>	38.4	41.2	<b>42.7</b>	42.4
XLNet-large-cased	37.3	39.7	<b>46.4</b>	34.1	40.6	41.1	<b>41.2</b>
RoBERTa-base	38.0	41.0	<b>45.0</b>	35.9	41.7	<b>42.1</b>	39.6
RoBERTa-large	33.8	38.6	<b>42.8</b>	30.2	33.1	<b>37.5</b>	35.7
GPT2	35.4	34.5	<b>38.5</b>	21.9	26.1	<b>27.2</b>	26.1
GPT2-medium	37.8	38.5	<b>39.8</b>	19.4	<b>29.1</b>	<b>29.1</b>	27.2
AVG	36.3	39.0	<b>43.3</b>	28.8	35.3	<b>36.9</b>	35.1
AVG w/o GPT2 *	36.2	39.8	<b>44.7</b>	31.6	37.9	<b>39.8</b>	38.0

Table 1: Unlabeled sentence-level parsing  $F_1$  scores on the English PTB test set. <sup>†</sup>: the best results of the top single head and layer-wise ensemble. <sup>‡</sup>: directly applying the chart-based parser for ranking (no development set trees) and ensembling the top- $K$  heads for parsing. \*: average  $F_1$  scores without GPT2 and GPT2-medium. Bold figures highlight the best scores for the two different groups: with and without development trees.

Model	$F_1$	SBAR	NP	VP	PP	ADJP	ADVP
Balanced	18.5	7	27	8	18	27	25
Left branching	8.7	5	11	0	5	2	8
Right branching	39.4	<b>68</b>	24	<b>71</b>	42	27	38
BERT-base-cased	37.1	36	49	30	42	40	69
BERT-large-cased	38.7	38	50	30	46	42	<b>72</b>
XLNet-base-cased	<b>42.7</b>	45	<b>58</b>	31	46	46	<b>72</b>
XLNet-large-cased	41.1	44	54	30	42	<b>48</b>	64
RoBERTa-base	42.1	38	<b>58</b>	31	<b>47</b>	42	71
RoBERTa-large	37.5	35	53	29	33	36	54

Table 2: Unlabeled parsing scores and recall scores on six constituency tags of trivial baseline parse trees as well as ones achieved by our parser using dynamic  $K$  on different PLMs.

### 5.2 Results on the English PTB

We first evaluate our ranking-based zero-shot parser on the English PTB dataset. We apply our methods to four different PLMs for English: BERT (Devlin et al., 2019), XLNet (Yang et al., 2019), RoBERTa (Liu et al., 2019b), and GPT2 (Radford et al., 2019).<sup>10</sup>

Table 1 shows the unlabeled  $F_1$  scores for our ranking-based zero-shot parser as well as for previous zero-shot parsers in two settings, with and without an annotated development set. We employ the chart-based parser in a setting without development trees, where Eqs. (1) and (2) are used for

<sup>10</sup>We follow previous work (Kim et al., 2020a,b) in using two variants for each PLM, where the X-base variants consist of 12 layers, 12 attention heads, and 768 hidden dimensions, while the X-large ones have 24 layers, 16 heads, and 1024 dimensions. With regard to GPT2, the GPT2 model corresponds to X-base while GPT2-medium to X-large.

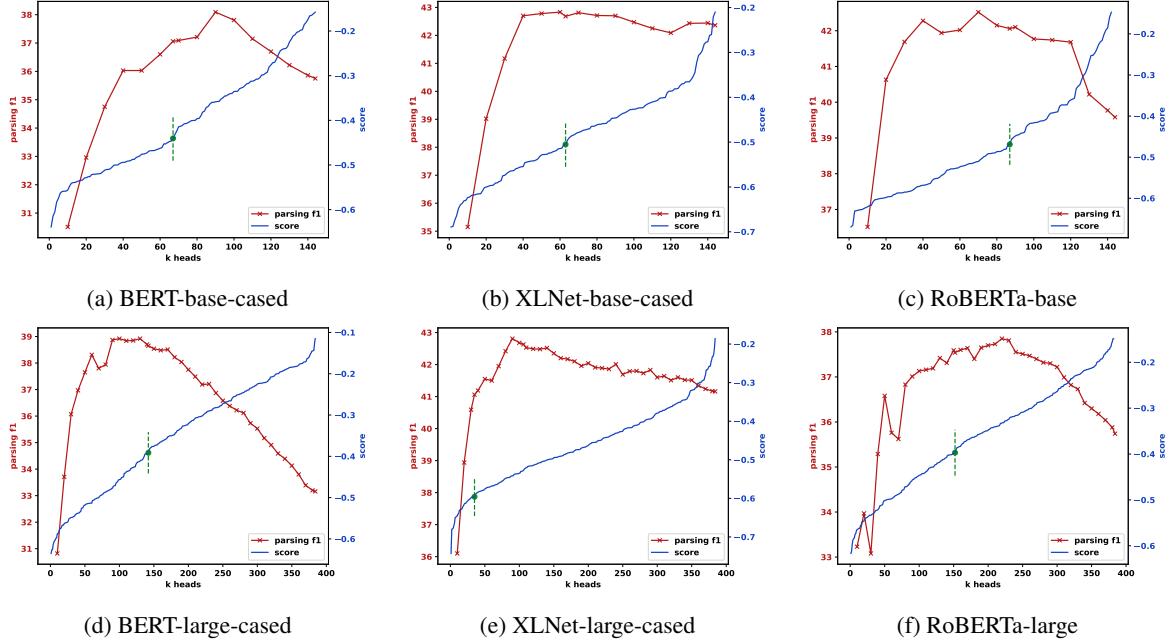


Figure 1: Relation between  $K$  for top- $K$  and parsing performance on different PLMs. The blue curve shows the ranking score of heads where heads are sorted in an ascending order. The red curve shows the parsing performance that is evaluated on the PTB test set given every 10 heads. The green dashed line indicates the dynamic  $K$ .

ranking and ensembling the top- $K$  (i.e., top-30) heads. Compared to our method under the same configuration, its poor performance confirms the effectiveness of our ranking algorithm.

With respect to the  $K$  selection, our dynamic  $K$  method beats both fixed top-30 and full heads. Surprisingly, using all attention heads for ensemble parsing yields nearly the same performance as using top-30 heads. This suggests that although our ranking algorithm filters out some noisy heads, it is still not perfect. On the other hand, the ensemble parsing method is robust to noisy heads when full attention heads are used. Figure 1 shows how the ensemble parsing performance changes given different  $K$  selection. We can identify a roughly concave shape of the parsing performance curve, which indicates why our ranking algorithm works. Interestingly, the parsing performance does not drop too much when  $K$  reaches the maximum for XLNet. We conjecture that syntactic knowledge is more broadly distributed across heads in XLNet.

Our ranking-based parser performs badly on GPT2 and GPT2-medium, which is not unexpected. Unlike other PLMs, models in the GPT2 category are auto-regressive language models, whose attention matrix is strictly lower triangular. It makes it hard for our ranking algorithm to work properly. But for top-down and chart-based zero-shot parsers, tuning against an annotated development set can

alleviate this problem. We focus on BERT, XLNet and RoBERTa and only evaluate these three models in the rest of our experiments. Except for GPT2 variants, our parser with dynamic  $K$  outperforms the top-down parser in all cases. On average (without GPT2 variants), even though our parser only requires raw sentence data, it still matches the chart-based parser with the top single head or layer-wise ensemble. To explore the limit of the chart-based parser, we also present the results by selecting the top- $K$  (i.e., top-20) heads using the annotated development set (Kim et al., 2020b).<sup>11</sup> Note that in this setting, the best configuration, i.e., the combination of  $g$ ,  $f$  and  $s_{comp}$  as well as  $K$  are selected against the development set. This setting serves as an upper bound of the chart-based zero-shot parsing and largely outperforms our ranking-based method.

Table 2 presents the parsing scores as well as recall scores on different constituents of trivial baselines and our parser. It indicates that trees induced from XLNet-base-cased, XLNet-large-cased and RoBERTa-base can outperform the right-branching baseline without resembling it. This confirms that PLMs can produce non-trivial parse trees. Large gains on NP, ADJP and ADVP compared to the

<sup>11</sup>Selecting heads against a development set ensures the quality of high ranking heads; top-20 heads are optimal in this setting (Kim et al., 2020b), unlike top-30 in our setting.

Model	English	Basque	French	German	Hebrew	Hungarian	Korean	Polish	Swedish	AVG
Trivial baselines										
Balanced	18.5	24.4	12.9	15.2	18.1	14.0	20.4	26.1	13.3	18.1
Left branching	8.7	14.8	5.4	14.1	7.7	10.6	16.5	28.7	7.6	12.7
Right branching	39.4	22.4	1.3	3.0	0.0	0.0	21.1	0.7	1.7	10.0
Chart-based (Single/Layer) <sup>†</sup>										
w/ dev trees	M-BERT	41.2	38.1	30.6	32.1	31.9	30.4	46.4	43.5	27.5
	XLM	43.0	35.3	35.6	41.6	39.9	34.5	35.7	51.7	33.7
	XLM-R	44.4	40.4	31.0	32.8	34.1	32.4	47.5	44.7	29.2
	XLM-R-large	40.8	36.5	26.4	30.2	32.1	26.8	45.6	47.9	25.8
	AVG	42.4	37.6	30.9	34.2	34.5	31.0	43.8	46.9	29.1
	Chart-based (top- $K$ ) <sup>†</sup>									
	M-BERT	45.0	41.2	35.9	35.9	37.8	33.2	47.6	51.1	32.6
	XLM	<b>47.7</b>	41.3	<b>36.7</b>	<b>43.8</b>	<b>41.0</b>	36.3	35.7	<b>58.5</b>	<b>36.5</b>
	XLM-R	47.0	<b>42.2</b>	35.8	37.7	40.1	<b>36.6</b>	<b>51.0</b>	52.7	32.9
	XLM-R-large	45.1	40.2	29.7	37.1	36.2	31.0	46.9	47.9	27.8
	AVG	46.2	41.2	34.5	38.6	38.8	34.3	45.3	52.6	32.5
w/o dev trees	Crosslingual ranking-based (Dynamic $K$ ) <sup>‡</sup>									
	M-BERT	40.7	<b>38.2</b>	31.0	31.0	29.0	27.1	43.3	30.7	25.8
	XLM	44.9	26.6	<b>35.8</b>	<b>39.7</b>	<b>39.6</b>	32.9	28.0	<b>50.1</b>	<b>34.1</b>
	XLM-R	<b>45.5</b>	<b>38.2</b>	34.0	35.5	36.7	<b>33.5</b>	<b>45.2</b>	39.4	29.9
	XLM-R-large	41.0	37.9	28.0	28.0	31.3	24.6	44.4	32.2	24.9
	AVG	43.0	34.7	32.4	33.5	35.0	29.8	40.4	39.2	29.2
	35.3									

Table 3: Parsing results on nine languages with multilingual PLMs. <sup>†</sup>: attention heads are selected on the development trees in the target language. <sup>‡</sup>: attention heads are selected on raw sentences in English. Bold figures highlight the best scores for the two different groups: with and without development trees.

right branching baseline show that PLMs can better identify such constituents.

### 5.3 Results for Languages other than English

Low-resource language parsing is one of the main motivations for the development of unsupervised parsing algorithms, which makes a multilingual setting ideal for evaluation. Multilingual PLMs are attractive in this setting because they are trained to process over one hundred languages in a language-agnostic manner. Kim et al. (2020b) has investigated the zero-shot parsing capability of multilingual PLMs assuming that a small annotated development set is available. Here, by taking advantage of our ranking-based parsing algorithm, we use a more radical crosslingual setting. We rank attention heads only on sentences in English and directly apply the parser to eight other languages. We follow Kim et al. (2020b) and use four multilingual PLMs: a multilingual version of the BERT-base model (M-BERT, Devlin et al. 2019), the XLM model (Conneau and Lample, 2019), the XLM-R and XLM-R-large models (Conneau et al., 2020). Each multilingual PLM differs in architecture and pre-training data, and we refer readers to the original papers for more details.

In Table 3, our crosslingual parser outperforms the trivial baselines in all cases by a large margin. Compared with the chart-based parser with the top head or layer-wise ensemble, our crosslingual parser can match the performance on five out of nine languages. Among four model variants, XLM-R and XLM-R-large have identical training settings and pre-training data, and so form a controlled experiment. By directly comparing XLM-R and XLM-R-large, we conjecture that, as the capacity of the PLM scales, the model has more of a chance to learn separate hidden spaces for different languages. This is consistent with a recent study on multilingual BERT (Dufter and Schütze, 2020) showing that underparameterization is one of the main factors that contribute to multilinguality. Again, our method lags behind the chart-base zero-shot parser with a top- $K$  ensemble. More experimental results including using target language for head selection in our method can be found in Appendix A.1.

### 5.4 Grammar Analysis

By not relying on an annotated development set, we have an unbiased way of investigating the tree structures as well as the grammars that are inher-

Trees	Preterminal Acc <sup>†</sup>	Rule Acc <sup>‡</sup>	Parsing <i>F</i> <sub>1</sub>
Gold*	66.1	<b>46.2</b>	-
BERT-base-cased	64.4	24.8	37.1
BERT-large-cased	64.0	22.3	38.7
XLNet-base-cased	<b>67.7</b>	26.1	42.7
XLNet-large-cased	65.8	27.3	41.1
RoBERTa-base	65.7	27.2	42.1
RoBERTa-large	62.4	25.1	37.5

Table 4: Preterminal (PoS tag) and production rule accuracies of PCFG<sub>PLM</sub> and PCFG<sub>Gold</sub> on the entire PTB. †: PoS tagging accuracy using the many-to-one mapping (Johnson, 2007). ‡: production rule accuracy where anonymized nonterminals and preterminals are mapped to the gold tags using the many-to-one mapping. \*: PCFG<sub>Gold</sub>.

ent in PLMs. Specifically, we first parse the raw sentences using our ranking-based parser described in Section 3.2 and then train a neural PCFG given the induced trees using the method in Section 4.2. We conduct our experiments on the English PTB and evaluate how the learned grammar resembles PTB syntax in a quantitative way on preterminals (PoS tags) and production rules. We visualize the alignment of preterminals and nonterminals of the learned grammar and the gold labels in Appendix A.2 as a qualitative study. We also showcase parse trees of the learned grammar to get a glimpse of some distinctive characteristics of the learned grammar in Appendix A.3. For brevity, we refer to a neural PCFG learned from trees induced of a PLM as PCFG<sub>PLM</sub> and to a neural PCFG learned from the gold parse trees as PCFG<sub>Gold</sub>.

In Table 4, we report preterminal (unsupervised PoS tagging) accuracies and production rule accuracies of PCFG<sub>PLM</sub> and PCFG<sub>Gold</sub> on the corpus level. For preterminal evaluation, we map the anonymized preterminals to gold PoS tags using many-to-one (M-1) mapping (Johnson, 2007), where each anonymized preterminal is matched onto the gold PoS tag with which it shares the most tokens. For production rule evaluation, we map both nonterminals and preterminals to gold tags using M-1 mapping to get the binary production rules.<sup>12</sup> We find that all PCFG<sub>PLM</sub> grammars except for PCFG<sub>RoBERTa-large</sub> outperform a discrete HMM baseline (62.7, He et al. 2018) but are far from the state of the art for neural grammar induction.

<sup>12</sup>For the gold annotations, we drop all unary rules. For  $n$ -ary rules ( $n > 2$ ), we convert them to binary rules by right branching and propagating the parent tag. For example, a  $n$ -ary rule  $A \rightarrow B C D$  yields  $A \rightarrow B A$  and  $A \rightarrow C D$ .

(80.8, He et al. 2018). All PCFG<sub>PLM</sub> produce similar accuracies on preterminals as PCFG<sub>Gold</sub>. However, for the production rules, PCFG<sub>PLM</sub> lags behind PCFG<sub>Gold</sub> by a large margin. This makes sense as presumably the tree structures heavily affect nonterminal learning. We also present the parsing  $F_1$  scores of corresponding trees against the gold trees in Table 4 for comparison. We observe that for all PCFG<sub>PLM</sub>, both preterminal accuracies and production rule accuracies correlate well with the parsing  $F_1$  scores of the corresponding trees.

## 6 Conclusion

In this paper, we set out to analyze the syntactic knowledge learned by transformer-based pre-trained language models. In contrast to previous work relying on test suites and probes, we proposed to use a zero-shot unsupervised parsing approach. This approach is able to parse sentences by ranking the attention heads of the PLM and ensembling them. Our approach is able to completely do away with a development set annotated with syntactic structures, which makes it ideal in a strictly unsupervised setting, e.g., for low resource languages. We evaluated our method against previous methods on nine languages. When development sets are available for previous methods, our method can match them or produce competitive results if they use the top single head or layer-wise ensembling of attention heads, but lags behind them if they ensemble the top- $K$  heads. Furthermore, we present an analysis of the grammars learned by our approach: we use the induced trees to train a neural PCFG and evaluate the pre-terminal and non-terminal symbols of that grammar. In future work, we will develop further methods for analyzing the resulting grammar rules. Another avenue for follow-up research is to use our method to determine how the syntactic structures inherent in PLMs change when these models are fine-tuned on a specific task.

## Acknowledgments

We thank the reviewers for their valuable suggestions regarding this work.

## References

- James K Baker. 1979. Trainable grammars for speech recognition. *The Journal of the Acoustical Society of America*.
- Yonatan Belinkov and James Glass. 2019. Analysis

- methods in neural language processing: A survey. *TACL*, 7:49–72.
- Glenn Carroll and Eugene Charniak. 1992. Two experiments on learning probabilistic dependency grammars from corpora. In *AAAI Workshop on Statistically-Based NLP Techniques*.
- Jihun Choi, Kang Min Yoo, and Sang-goo Lee. 2018. Learning to compose task-specific tree structures. In *AAAI*.
- Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does bert look at? an analysis of bert’s attention. In *Black-BoxNLP@ACL*.
- John Cocke. 1969. *Programming languages and their compilers: Preliminary notes*.
- Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *ACL*.
- Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *NeurIPS*.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.
- Andrew Drozdov, Patrick Verga, Mohit Yadav, Mohit Iyyer, and Andrew McCallum. 2019. Unsupervised latent tree induction with deep inside-outside recursive auto-encoders. In *NAACL*.
- Philipp Dufter and Hinrich Schütze. 2020. Identifying necessary elements for bert’s multilinguality. *arXiv preprint arXiv:2005.00396*.
- Yoav Goldberg. 2019. Assessing BERT’s syntactic abilities. *arXiv preprint arXiv:1901.05287*.
- Junxian He, Graham Neubig, and Taylor Berg-Kirkpatrick. 2018. Unsupervised learning of syntactic structure with invertible neural projections. In *EMNLP*.
- John Hewitt and Percy Liang. 2019. Designing and interpreting probes with control tasks. In *EMNLP-IJCNLP*.
- John Hewitt and Christopher D Manning. 2019. A structural probe for finding syntax in word representations. In *NAACL*.
- Phu Mon Htut, Kyunghyun Cho, and Samuel Bowman. 2018. Grammar induction with neural language models: An unusual replication. In *BlackboxNLP@EMNLP*.
- Mark Johnson. 2007. Why doesn’t em find good hmm pos-taggers? In *EMNLP-CoNLL*.
- Katharina Kann, Kyunghyun Cho, and Samuel R Bowman. 2019. Towards realistic practices in low-resource natural language processing: The development set. In *EMNLP-IJCNLP*.
- Tadao Kasami. 1966. An efficient recognition and syntax-analysis algorithm for context-free languages. *Coordinated Science Laboratory Report no. R-257*.
- Taeuk Kim, Jihun Choi, Daniel Edmiston, and Sang-goo Lee. 2020a. Are pre-trained language models aware of phrases? simple but strong baselines for grammar induction. In *ICLR*.
- Taeuk Kim, Bowen Li, and Sang-goo Lee. 2020b. Multilingual zero-shot constituency parsing. *arXiv preprint arXiv:2004.13805v2*.
- Yoon Kim, Chris Dyer, and Alexander Rush. 2019a. Compound probabilistic context-free grammars for grammar induction. In *ACL*.
- Yoon Kim, Alexander Rush, Lei Yu, Adhiguna Kunoro, Chris Dyer, and Gábor Melis. 2019b. Unsupervised recurrent neural network grammars. In *NAACL*.
- Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of BERT. In *EMNLP-IJCNLP*.
- Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019a. Linguistic knowledge and transferability of contextual representations. In *NAACL*.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. In *arXiv preprint arXiv:1907.11692*.
- Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*.
- David Mareček and Rudolf Rosa. 2018. Extracting syntactic trees from transformer encoder self-attentions. In *BlackboxNLP@EMNLP*.
- David Mareček and Rudolf Rosa. 2019. From balustrades to pierre vinken: Looking for syntax in transformer self-attentions. In *BlackboxNLP@ACL*.
- Rowan Hall Maudslay, Josef Valvoda, Tiago Pimentel, Adina Williams, and Ryan Cotterell. 2020. A tale of a probe and a parser. In *ACL*.
- Tiago Pimentel, Josef Valvoda, Rowan Hall Maudslay, Ran Zmigrod, Adina Williams, and Ryan Cotterell. 2020. Information-theoretic probing for linguistic structure. In *ACL*.

- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. [Language models are unsupervised multitask learners](#).
- Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. [A primer in bertology: What we know about how bert works](#). *arXiv preprint arXiv:2002.12327*.
- Djamé Seddah, Reut Tsarfaty, Sandra Kübler, Marie Candito, Jinho D. Choi, Richárd Farkas, Jennifer Foster, Iakes Goenaga, Koldo Gojenola Galletebeitia, Yoav Goldberg, Spence Green, Nizar Habash, Marco Kuhlmann, Wolfgang Maier, Joakim Nivre, Adam Przeździecki, Ryan Roth, Wolfgang Seeker, Yannick Versley, Veronika Vincze, Marcin Woliński, Alina Wróblewska, and Eric Villemonte de la Clergerie. 2013. [Overview of the SPMRL 2013 shared task: A cross-framework evaluation of parsing morphologically rich languages](#). In *4th Workshop on Statistical Parsing of Morphologically-Rich Languages*.
- Yikang Shen, Zhouhan Lin, Chin wei Huang, and Aaron Courville. 2018a. [Neural language modeling by jointly learning syntax and lexicon](#). In *ICLR*.
- Yikang Shen, Zhouhan Lin, Athul Paul Jacob, Alessandro Sordoni, Aaron Courville, and Yoshua Bengio. 2018b. [Straight to the tree: Constituency parsing with neural syntactic distance](#). In *ACL*.
- Yikang Shen, Shawn Tan, Alessandro Sordoni, and Aaron Courville. 2019. [Ordered neurons: Integrating tree structures into recurrent neural networks](#). In *ICLR*.
- Haoyue Shi, Karen Livescu, and Kevin Gimpel. 2020. [On the role of supervision in unsupervised constituency parsing](#). In *EMNLP*.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. [Attention is all you need](#). In *NeurIPS*.
- Jesse Vig. 2019. [A multiscale visualization of attention in the transformer model](#). In *ACL*.
- Elena Voita and Ivan Titov. 2020. [Information-theoretic probing with minimum description length](#). *arXiv preprint arXiv:2003.12298*.
- Adina Williams, Andrew Drozdov, and Samuel R. Bowman. 2018. [Do latent tree learning models identify meaningful structure in sentences?](#) *TACL*, 6:253–267.
- Zhiyong Wu, Yun Chen, Ben Kao, and Qun Liu. 2020. [Perturbed masking: Parameter-free probing for analyzing and interpreting bert](#). In *ACL*.
- Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. [Xlnet: Generalized autoregressive pretraining for language understanding](#). In *NeurIPS*.
- Daniel H Younger. 1967. Recognition and parsing of context-free languages in time  $n^3$ . *Information and control*.
- Kelly Zhang and Samuel Bowman. 2018. [Language modeling teaches you more than translation does: Lessons learned through auxiliary syntactic task analysis](#). In *BlackboxNLP@EMNLP*.
- Hao Zhu, Yonatan Bisk, and Graham Neubig. 2020. [The return of lexical dependencies: Neural lexicalized PCFGs](#). *TACL*.

## A Appendix

### A.1 More Results on Languages other than English

We present a comprehensive analysis of the chart-based parser and our ranking-based parser on the multilingual setting. In addition to Table 3, for our method, we conduct experiments using target language for head selection with both Top- $K$  (i.e., top-30) ensemble and dynamic  $K$  ensemble.

In Table 5, we find that our ranking-based parser with Top- $K$  ensemble performs slightly better than that using dynamic  $K$ . In contrast to the superiority of dynamic  $K$  on English PLMs in Table 1, multilingual PLMs produce similar parsing performance with a *lazy* top-30 ensemble. We conjecture that there could be no clear concave pattern (like Figure 1) in the relation of  $K$  and parsing performance in this crosslingual setting.

We also experimented with another setting for our ranking-based parser: selecting attention heads based on the sentences in the target language. Interestingly, we observe a considerable parsing performance drop on both top- $K$  and dynamic  $K$  ensemble. We suspect that our chart-based ranking algorithm (e.g., the inherent context free grammar assumption) does not work equally well in all languages, at least for the annotation scheme provided by the SPMRL dataset. In this scenario, using English for head selection has a better chance to capture syntax-related attention heads. Again, as we discussed before, using annotated trees in the target language can always ensure the quality of selected top- $K$  heads.

### A.2 Visualization of the Alignment for Internal Tags

Since the recall scores in Table 2 have shown ability of PLMs to identify different nonterminals, here we visualize the alignment between PCFG internal tags and corresponding gold labels in Figures 2 and 3. For the nonterminal alignment, some of the learned nonterminals clearly align to gold standard labels, in particular for frequent ones like NP and VP. Compared to PCFG<sub>Gold</sub>, PCFG<sub>PLM</sub> learns a more uncertain grammar and resulting in overall lower precision.

But for the preterminal (PoS tag) alignment, no clear difference can be identified between PCFG<sub>Gold</sub> and PCFG<sub>PLM</sub>. This is consistent with the finding in Table 4 that all PCFG<sub>PLM</sub> produce similar accuracies on preterminals as PCFG<sub>Gold</sub>.

### A.3 Parse tree samples

In Figure 4, we show parse trees obtained by PCFG<sub>Gold</sub>, PCFG<sub>PLM</sub> and the gold standard reference on a sample sentence. In this sample, PCFG<sub>Gold</sub> predicts the constituency tree structure accurately. On the development set, PCFG<sub>Gold</sub> reaches around 72 unlabeled  $F_1$  score, as it is supervised by the PTB trees. Although this is a low  $F_1$ -score, it is not untypical for PCFG-based models, which are limited by their insufficiently flexible rules and their lack of lexicalization. Also note that the oracle trees only yield 84.3  $F_1$ . PCFG<sub>PLM</sub> perform worse than PCFG<sub>Gold</sub> when compared against the gold tree. They are able to identify short NPs, but don't work well for larger constituents. We also observe some frequent incorrect patterns which are also present in this example, e.g., grouping VBD with the preceding NP, or IN with the preceding VBD.

Language	English	Basque	French	German	Hebrew	Hungarian	Korean	Polish	Swedish	Avg
Trivial baselines										
Balanced	18.5	24.4	12.9	15.2	18.1	14.0	20.4	26.1	13.3	18.1
Left branching	8.7	14.8	5.4	14.1	7.7	10.6	16.5	28.7	7.6	12.7
Right branching	39.4	22.4	1.3	3.0	0.0	0.0	21.1	0.7	1.7	10.0
Chart-based (Single/Layer) <sup>†</sup>										
M-BERT	41.2	38.1	30.6	32.1	31.9	30.4	46.4	43.5	27.5	35.7
XLM	43.0	35.3	35.6	41.6	39.9	34.5	35.7	51.7	33.7	39.0
XLM-R	44.4	40.4	31.0	32.8	34.1	32.4	47.5	44.7	29.2	37.4
XLM-R-large	40.8	36.5	26.4	30.2	32.1	26.8	45.6	47.9	25.8	34.7
AVG	42.4	37.6	30.9	34.2	34.5	31.0	43.8	46.9	29.1	36.7
Chart-based (Top- $K$ ) <sup>†</sup>										
M-BERT	45.0	41.2	35.9	35.9	37.8	33.2	47.6	51.1	32.6	40.0
XLM	<b>47.7</b>	41.3	<b>36.7</b>	<b>43.8</b>	<b>41.0</b>	36.3	35.7	<b>58.5</b>	<b>36.5</b>	<b>41.9</b>
XLM-R	47.0	<b>42.2</b>	35.8	37.7	40.1	<b>36.6</b>	<b>51.0</b>	52.7	32.9	41.8
XLM-R-large	45.1	40.2	29.7	37.1	36.2	31.0	46.9	47.9	27.8	38.0
AVG	46.2	41.2	34.5	38.6	38.8	34.3	45.3	52.6	32.5	40.4
Ranking-based (Top- $K$ ) <sup>‡</sup>										
M-BERT	41.5	38.9	33.9	30.2	36.3	30.9	39.0	18.4	26.3	31.7
XLM	44.6	21.0	29.8	39.2	30.5	25.2	23.8	55.2	30.3	31.9
XLM-R	44.8	36.0	34.1	31.8	36.4	32.5	40.3	29.6	26.7	33.4
XLM-R-large	41.1	36.8	30.3	26.8	33.4	24.9	37.4	17.5	26.3	29.2
AVG	43.0	33.2	32.0	32.0	34.2	28.4	35.1	30.2	27.4	31.6
Ranking-based (Dynamic $K$ ) <sup>‡</sup>										
M-BERT	40.7	39.1	28.4	25.5	26.9	31.2	41.3	22.2	21.3	29.5
XLM	44.9	20.8	29.9	40.3	34.4	27.7	23.6	55.1	31.2	32.9
XLM-R	45.5	37.3	30.7	31.5	31.8	34.1	40.8	36.0	27.4	33.7
XLM-R-large	41.0	36.5	29.0	30.1	32.6	25.3	43.9	30.0	25.5	31.6
AVG	43.0	33.4	29.5	31.9	31.4	29.6	37.4	35.8	26.4	31.9
Crosslingual ranking-based (Top- $K$ ) <sup>‡</sup>										
M-BERT	-	37.9	33.4	31.2	31.5	29.4	45.3	33.4	27.2	34.5
XLM	-	25.9	34.4	39.2	<b>39.5</b>	31.9	27.5	<b>50.4</b>	<b>34.2</b>	36.4
XLM-R	-	37.9	33.9	35.1	36.8	33.3	44.7	39.7	30.3	37.4
XLM-R-large	-	35.7	28.5	28.5	34.7	25.5	44.5	36.9	27.1	33.6
AVG	-	34.3	32.6	33.5	35.6	30.0	40.5	40.1	29.7	35.5
Crosslingual ranking-based (Dynamic $K$ ) <sup>‡</sup>										
M-BERT	-	<b>38.2</b>	31.0	31.0	29.0	27.1	43.3	30.7	25.8	33.0
XLM	-	26.6	<b>35.8</b>	<b>39.7</b>	<b>39.6</b>	32.9	28.0	50.1	34.1	36.9
XLM-R	-	<b>38.2</b>	34.0	35.5	36.7	<b>33.5</b>	<b>45.2</b>	39.4	29.9	<b>37.6</b>
XLM-R-large	-	37.9	28.0	28.0	31.3	24.6	44.4	32.2	24.9	32.5
AVG	-	34.7	32.4	33.5	35.0	29.8	40.4	39.2	29.2	35.3

Table 5: Parsing results on nine languages with multilingual PLMs. Except for the trivial baselines, all experimental results are divided into two groups: using target language for head selection and using English for head selection (crosslingual). <sup>†</sup>: results of the best configurations of  $f$ ,  $g$ ,  $s_{comp}$  and  $K$  are decided on an annotated development set. <sup>‡</sup>: results where only raw sentences are required. For top- $K$ , 20 is used for chart-based and 30 is used for our ranking-based. Bold figures highlight the best scores for the two different groups: using target language and English for head selection.

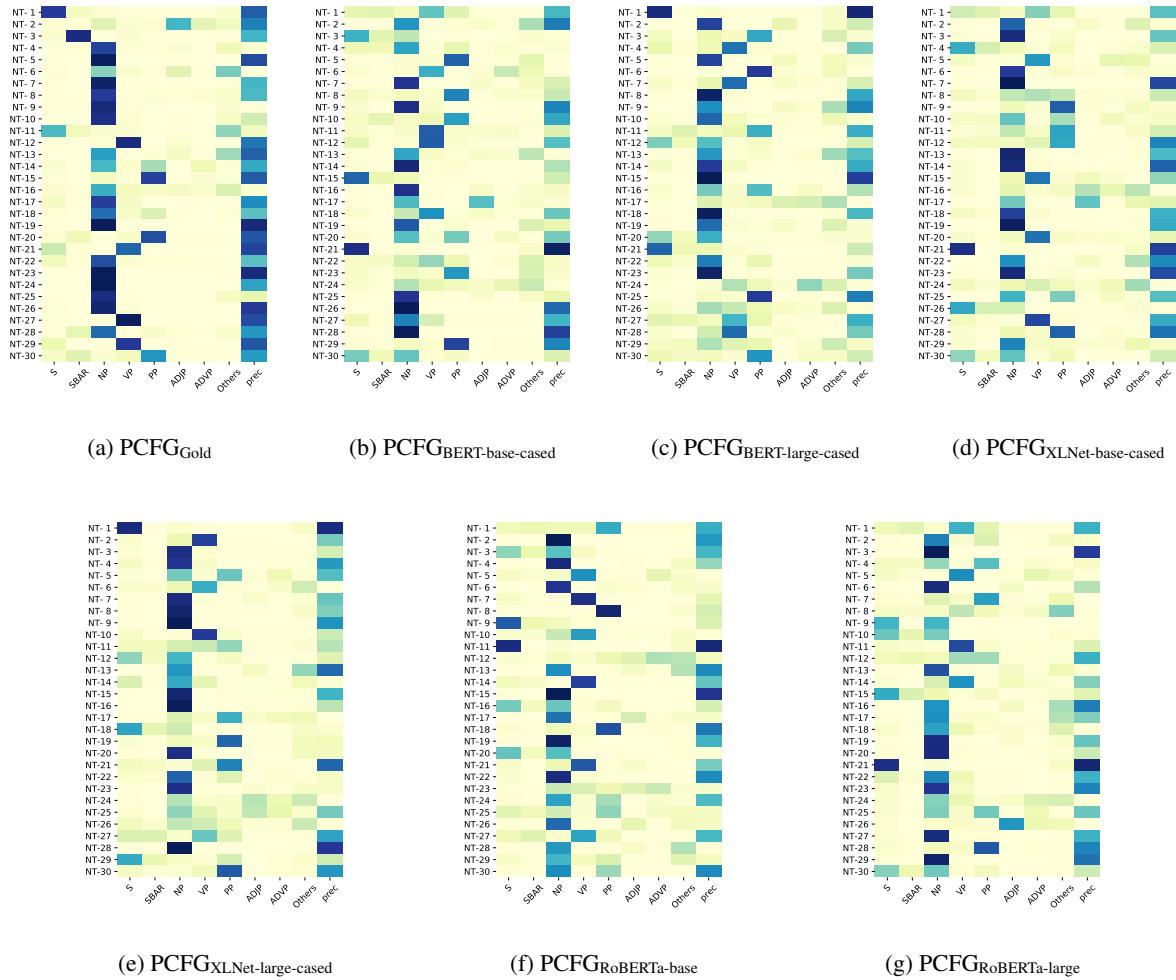
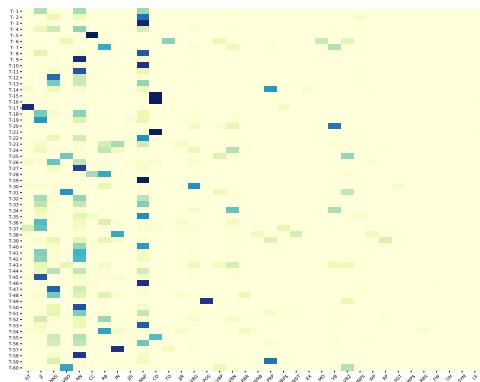
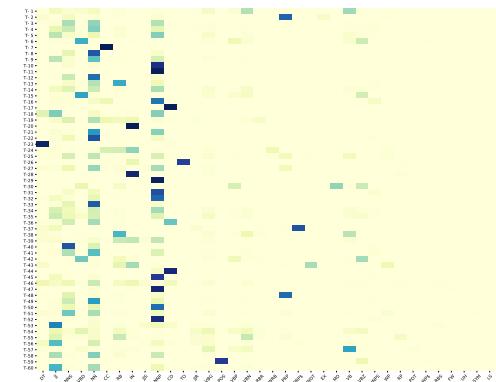


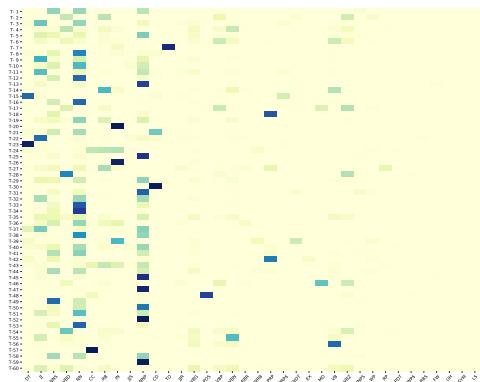
Figure 2: Alignment of induced nonterminals of PCFG<sub>PLM</sub> and PCFG<sub>Gold</sub> on the entire PTB. The last column prec shows the precision that a nonterminal predicts a particular gold constituent.



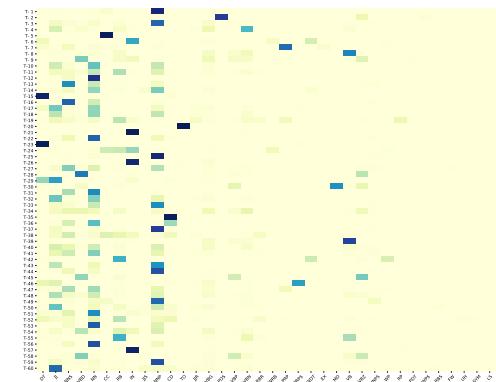
(a) PCFG<sub>Gold</sub>



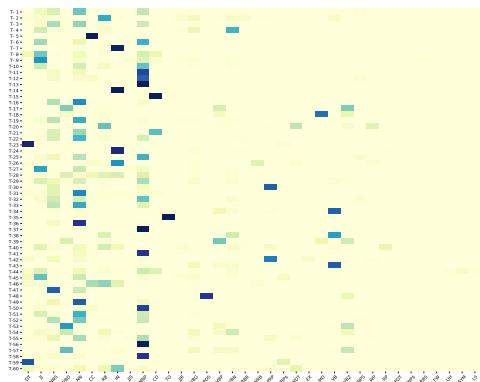
(b) PCFG<sub>BERT-base-cased</sub>



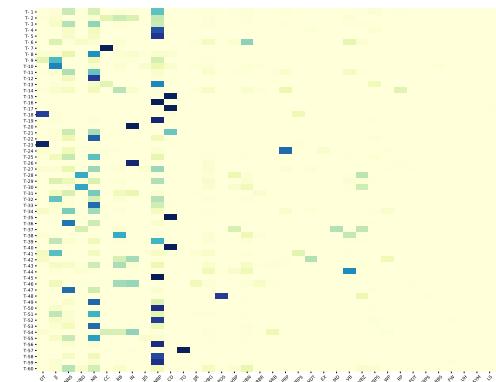
(c) PCFG<sub>BERT-large-cased</sub>



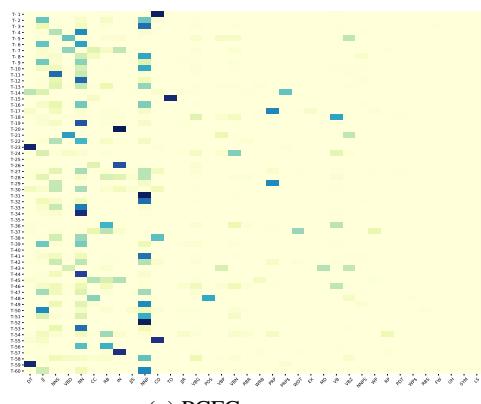
(d) PCFG<sub>XLNet-base-cased</sub>



(e) PCFG<sub>XLNet-large-cased</sub>



(f) PCFG<sub>RoBERTa-base</sub>



(g) PCFG<sub>RoBERTa-large</sub>

Figure 3: Alignment of induced preterminals (PoS tags) of PCFG<sub>PLM</sub> and PCFG<sub>Gold</sub> on the entire PTB. 423

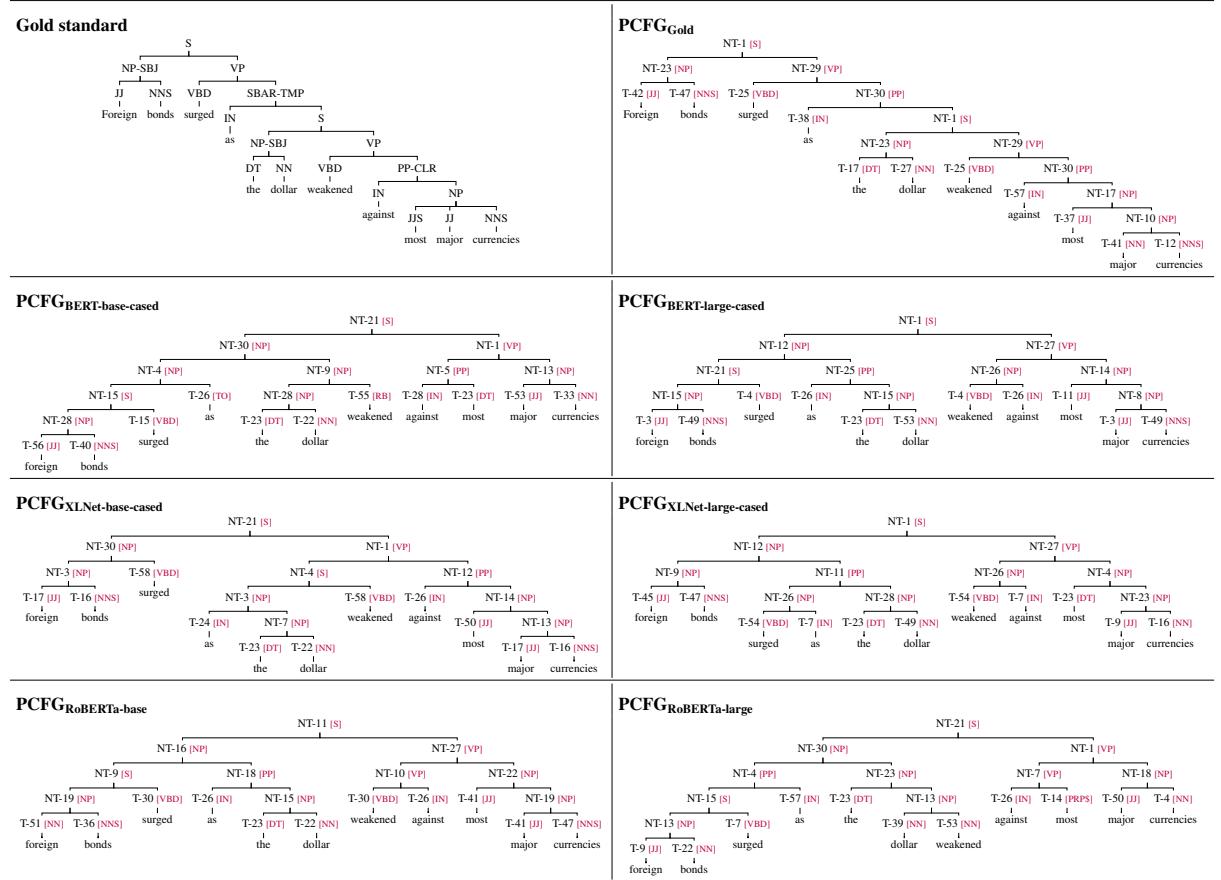


Figure 4: Parse tree samples of gold standard, PCFG<sub>Gold</sub>, and PCFG<sub>PLM</sub>. The mapped tag (marked in red) for each anonymized nonterminal and preterminal is obtained via many-to-one mapping.

# Building Location Embeddings from Physical Trajectories and Textual Representations

Laura Biester, Carmen Banea, Rada Mihalcea

Computer Science & Engineering, University of Michigan, USA

{lbiester, carmennb, mihalcea}@umich.edu

## Abstract

Word embedding methods have become the de-facto way to represent words, having been successfully applied to a wide array of natural language processing tasks. In this paper, we explore the hypothesis that embedding methods can also be effectively used to represent spatial locations. Using a new dataset consisting of the location trajectories of 729 students over a seven month period and text data related to those locations, we implement several strategies to create location embeddings, which we then use to create embeddings of the sequences of locations a student has visited. To identify the surface level properties captured in the representations, we propose a number of probing tasks such as the presence of a specific location in a sequence or the type of activities that take place at a location. We then leverage the representations we generated and employ them in more complex downstream tasks ranging from predicting a student’s area of study to a student’s depression level, showing the effectiveness of these location embeddings.

## 1 Introduction

Due to the rising adoption of smartphones over the past decade, the number of services with full or partial information about people’s spatial mobility has skyrocketed. Inspired by the natural language processing (NLP) literature, we investigate various properties of location embeddings. We explore whether valuable information is encoded in individual location embeddings, as well as embeddings that encompass a sequence of locations. We begin by exploring whether they are able to represent aspects such as location presence or location functionality. Ultimately, we test the hypothesis that if enough underlying information is encoded, embedding models should aid in predicting user-centered descriptors, such as area of study, academic status, or mental health.

Location data can be used by university administrators for applications that improve student life. From the frequency and the type of locations accessed in one’s daily routine, we may be able to identify someone who is depressed or someone who is overworked. Importantly, opt-in frameworks can be established to supplement existing counseling and advising offices, allowing for early intervention in the case of mental health and academic concerns. With proper privacy safeguards in place, such models could readily be applied on most university campuses, as WiFi connection data (from which we infer location) is likely already available. In addition, universities could use this data in an aggregate form to better understand student life and well-being, and find ways to promote healthy and engaging behaviors on campus. Such aggregate location information can also be used by architectural firms or municipalities to help with the selection of buildings’ locations, architecture, and design; with road and pedestrian traffic optimization; or for emergency response.

We also know that such data is already available to large technology companies that track their users, and it is important to spread awareness about the personal information that can be gleaned. Research like ours helps inform users about privacy concerns, and may open up a path to stricter legislation regarding the use of such data in the future. While we envision numerous positive applications of these methods, there are clear privacy drawbacks that the public should be aware of in the current technological environment.

Our work focuses on building an understanding of what information is encoded in location embeddings. In addition to creating embeddings using location trajectories, we propose an alternative method that synthesizes text from online sources to build representations that we hypothesize will better encode certain properties of locations. We

show that using dense location embeddings that incorporate both movement patterns and text data improves our ability to model downstream tasks. We see that although we are not able to recover as much surface level information from embeddings of location sequences as we are from a simpler representation, the additional semantic information that is encoded allows us to better predict some user attributes.

## 2 Related Work

**Embedding Evaluation and Probing.** Word embeddings are now widely used to create word representations using methods such as word2vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014), ELMo (Peters et al., 2018), and BERT (Devlin et al., 2019). BERT and ELMo can be used to create contextualized word embeddings, in which the vector representing an individual word varies depending on the context in which it appears. Previous methods including word2vec and GloVe did not make this distinction; adding context helped BERT achieve state-of-the-art results on many downstream NLP tasks. One traditional benchmark for word embeddings is performance on synthetic tasks, such as word similarity and word analogy tasks (Mikolov et al., 2013; Pennington et al., 2014). However, word embeddings are widely used because of their superior performance on a variety of downstream NLP tasks when compared to other word representations. Performance on downstream tasks has been used to evaluate sentence embeddings, however such approaches cannot gauge the content that is actually captured in the embeddings. To systematically ascertain what information is encoded in sentence vectors, researchers have turned to probing tasks (Shi et al., 2016; Adi et al., 2017; Conneau et al., 2018). These are meant to address the question “what information is encoded in a sentence vector” at a higher level.

In our work, we find inspiration in the research by Conneau et al. (2018), who propose a formalized evaluation technique for sentence embeddings using a suite of ten classification tasks focusing on: (1) surface information (e.g., length, word content), (2) syntactic information (e.g., bigram shift, tree depth), and (3) semantic information (e.g., tense). The deep learning methods gave the best results overall, but the bag-of-vectors approach was a solid baseline for the word content task, where it outper-

formed the deep learning models.

### Applications of Embeddings for Location Data.

Liu et al. (2016) were among the first to use the skip-gram model on location data. They use locations visited before and after a target location as context to create location embeddings. These are then used in a personalized location recommendation system. Feng et al. (2017) similarly create embeddings of check-in data, but use the CBOW model. Their application task is reversed, predicting future visitors for a location instead of predicting locations that a user will visit. Chang et al. (2018) also predict next check-ins for users using a model based on skip-gram. Their work is uniquely related to ours in that they also build prediction of the text content of check-ins into the objective function. Zhu et al. (2019) trained a skip-gram model to build location embeddings, and use them to understand the flow between urban locations. Crivellari and Beinat (2019) explore location embeddings from the perspective of geoinformatics, paving the way for our probing tasks.

The work of Solomon et al. (2018) is most similar to our own. They use GPS data from cell phones as input to create embeddings and use data from a university setting. Our work differs in that we use the skip-gram model and incorporate text-based embeddings. We also propose probing tasks to better understand the embeddings that we create, and predict additional user attributes from our new dataset that go beyond demographic information.

## 3 Data

### 3.1 Student and Location Data

Our dataset consists of location data collected from 729 undergraduate university students who agreed to participate in our study in 2018 and 2019 over a period of seven months.<sup>1</sup> Two-thirds of the students participated during the winter semester, and the other third during the fall semester. Dataset statistics are presented in Table 1.

Due to the sensitivity and scope of the data, it is infeasible for our study to include other universities; nonetheless, we believe that similar patterns would hold on other campuses as well. Because of privacy concerns, we are not able to publicly release this dataset.

<sup>1</sup>The data was collected as part of a study that underwent a full board review and was approved by the IRB at the University of Michigan (study number HUM00126298). All participants in the study have signed an informed consent form.

Number of Participants	729
Valid Location Visits After Pre-Processing	478,329
Unique Locations	194
Mean Locations per Participant	656.2
Mean Locations per Day	4.7

Table 1: Statistical summary of the location dataset.

While most similar research uses GPS (Solomon et al., 2018), mobile check-ins (Feng et al., 2017; Liu et al., 2016), or cell phone pings (Zhu et al., 2019) for location tracking, we collect location data from WiFi access logs. WiFi access logs provide a strong and unbiased location signal on campus, as most students carry their smart phones with them at all times; however, a downside is that we do not have location data for large time chunks when students are not connected to the campus WiFi.

The original data consists of 20,766,750 WiFi session updates across all the students. We only consider connections with uninterrupted updates from a single building (without a connection to a network in another building) for at least ten minutes. This ensures that a student’s location will not be mapped to multiple points during overlapping time spans, and that locations where a student does not spend a notable amount of time are excluded. After collecting this list of locations, start, and stop times, we perform a merging operation on the data, sorted by start time. If spans for the same location occur consecutively in the series with start and stop times less than 30 minutes apart, those spans are merged together.

After this pre-processing, we are left with 478,329 valid location spans with start and stop times. Since our dataset covers a single campus (194 locations), each location was manually labeled with its functionality, for a total of thirteen functionalities. The five most frequent are: class, study, dorm, lab, and library. While there are 194 locations in the location dataset, we utilize 132 in our analysis because this set of locations appears in all of the text-based datasets (described in Section 3.2); the ones that are left out are not among the most frequently visited.

In addition to location data, we collected a rich dataset containing information about the 729 students, consisting of a series of extensive surveys taken by the students throughout the semester and academic data from the registrar. From the survey data, we use information on class year, gender, depression, and sleep satisfaction. From the

Dataset	Campus Website	Reddit	Twitter
Overall Tokens	581K	882K	655K
Unique Tokens GloVe	9K	11K	18K
Median Instances Per Loc.	3.5	20.0	166.5
Start Date (year-month)	N/A	2011-05	2010-09
End Date (year-month)	2019-05	2019-07	2019-08

Table 2: Statistical information about text datasets.

academic data, we utilize the GPA and the school where the student is enrolled. These combined data sources are used for our downstream classification tasks. We chose students for the study covering all undergraduate class years, genders, and academic disciplines.

### 3.2 Text Data

In addition to location trajectories, we use text data from three sources (campus website, Reddit, Twitter) that illustrate various ways in which text can be used to represent places. Statistics of the text datasets are shown in Table 2.

**Campus Website.** With this dataset, we capture how people *formally define* locations. The university hosts a building search website that links to pages containing information about campus buildings, including the departments hosted inside. We manually link the locations in our dataset with building pages on this site, then scrape the first Google search result constrained within the university domain for each listed department, and use that text to represent the location. In addition to the departments, some pages directly link to a website (e.g., a gym links to recreational sports), from which we also scrape text.

**Reddit.** With this dataset, we capture how people *informally discuss* locations. From the university Reddit page, we search for building names. We increase the search term list using OpenStreetMap,<sup>2</sup> which lists alternate names for many buildings. We include text from posts and comments that specifically mention a building.

**Twitter.** With this dataset, we capture how people *express themselves* in various locations. We collect tweets that have been geotagged with GPS points within 0.05 kilometers of campus buildings.

<sup>2</sup><https://www.openstreetmap.org/>

## 4 Representing Locations

We use location trajectories and text data to create vector representations of locations and, subsequently, embeddings of sequences of locations that are visited by a single person. After pre-processing using the method described in Section 3.1, the location input data consists of a series of sorted, non-overlapping locations for a number of users with start and end times. We discuss multiple methods to create vector representations based on this data.

### 4.1 Location Trajectory-Based Representations

To create embeddings of locations, we make use of the temporal nature of the location trajectories to create a sequence of names of locations visited by a user over a period of time (e.g., the seven month period of our data collection, see Section 3.1). A skip-gram model is trained to use a location to predict locations around it in a user’s schedule, creating location embeddings that we expect will encode semantic information about locations.<sup>3</sup>

We represent each hour during the data collection period as a distinct token in the input trajectories. If a user has visited a single location in one hour, that location will be used in the slot for the hour; if they visited multiple locations, their predominant location will be used. If we do not have any location data for the user during that hour, we use the EXTERNAL token. This approach gives an exact meaning to the distance between locations in a sequence, while a raw sequence would ignore gaps in the data. The approach of using one token per set time interval is also used in Zhu et al. (2019). We refer to the method as Loc2V, and show a visualization in Figure 1.

### 4.2 Text-Based Representations

In addition to creating location representations from trajectories in the physical world, we explore the idea of using relevant text to define locations. Such text can reveal information about locations that may not be discernible from location trajectories, e.g. that people meet friends in a certain place. Therefore, for the same locations that appear in

<sup>3</sup>We use the default window size of 5 and generate embeddings with 25 dimensions. While 25 dimensions is fairly small in the context of word embeddings, since our dataset has fewer than two hundred locations that we seek to embed, higher values cannot be considered as leading to a dimensionality reduction. We use a negative sampling value of 20, as is suggested by Mikolov et al. (2013) for small datasets.

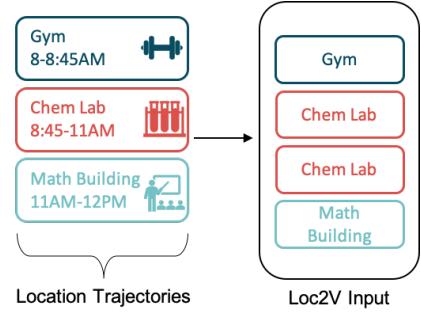


Figure 1: A sample sequence of locations, and the corresponding sequences that are used as Loc2V input.

our trajectories, we collect textual data that enables us to derive text-based representations from three sources as described in Section 3.2.

Using each textual data source, we map a location name to a set of relevant words. We calculate tf.idf (Salton and Buckley, 1988) weights for each word, then use those weights to compute a weighted average of pre-trained word embeddings. Because our datasets are primarily from social media, we use pre-trained GloVe embeddings that were obtained from Twitter data.<sup>4</sup> The resulting vector is used as a location representation.

### 4.3 Combining Representations

We hypothesize that trajectory based and text-based representations may encode different aspects of locations. Therefore, in addition to representing locations using text and physical trajectories, we experiment with combining the two. Our first method concatenates embedding vectors created from physical trajectories and vectors created from text data. Our second method performs retrofitting on top of text-based vectors. In the context of embeddings, “retrofitting” describes the process of modifying vectors that have already been created to better encode additional criteria. We find inspiration in the method from Faruqui et al. (2015), which retrofits word embeddings to a graph representing a semantic lexicon. In our work, we retrofit text embeddings to the graph that represents the transitions between locations; the nodes are locations, and the edges are weighted by the number of times there was a transition between those two locations in our dataset.

The retrofitting method takes a matrix  $\hat{Q}$ , the initial vectors, and updates matrix  $Q$  (initialized to  $\hat{Q}$ ) using a location transition graph. The objective

<sup>4</sup><https://nlp.stanford.edu/projects/glove/>

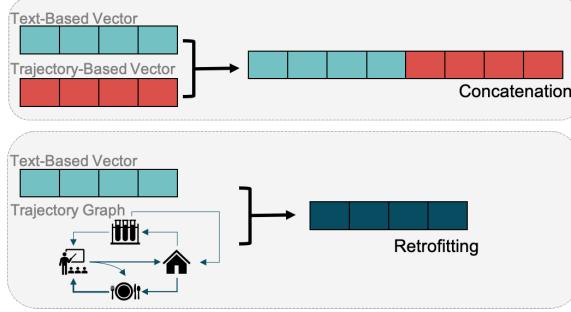


Figure 2: Comparison of the concatenation and retrofitting methods.

function incorporates the set of edges  $E$ , bringing vectors that share an edge closer together in the vector space:

$$\Psi(Q) = \sum_{i=1}^n \left[ \alpha_i \|q_i - \hat{q}_i\|^2 + \sum_{(i,j) \in E} \beta_{ij} \|q_i - q_j\|^2 \right]$$

An iterative method is used to update matrix  $Q$ :

$$q_i = \frac{\sum_{j:(i,j) \in E} \beta_{ij} q_j + \alpha_i \hat{q}_i}{\sum_{j:(i,j) \in E} \beta_{ij} + \alpha_i}$$

We perform ten iterations, as was done in previous work. The parameters  $\alpha$  and  $\beta$  control the relative importance of the two components (initial vectors and location graph). In their implementation, Faruqui et al. set  $\alpha_i = 1$  and  $\beta_{ij} = \text{degree}(i)^{-1}$ . As the graph we use is weighted, we introduce a weighted version that incorporates edge weights  $W$ , using a weighted inverse degree for  $\beta$ .

The retrofitting method enhances the text-based information by adding the assumption that locations that are visited sequentially are similar (in the sense that a person who visits one would visit the other), bringing them closer in the vector space. This method aims to infuse the text-based representations with information related to the co-occurrence of locations in a student’s trajectory; locations that co-occur may be suggestive of, for instance, areas of campus that tend to be visited by engineering students. It is not used on the trajectory-based representations, as these already incorporate location transitions.

Figure 2 compares the concatenation and retrofitting methods. As outlined above, the concatenation method directly combines the two vectors into one with the same content, while the retrofitting method takes information from a graph structure representing trajectories into account to create a modified version of the original vector.

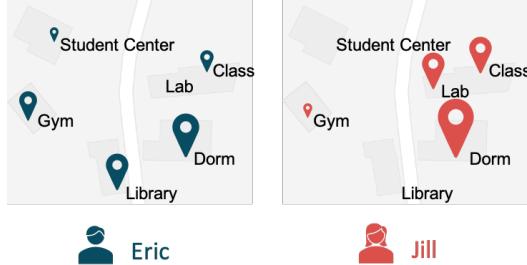


Figure 3: Fictional examples of locations visited by students; a larger pin reflects more time spent at a location.

#### 4.4 Representing a Sequence of Locations

To represent a sequence of locations, we use a vector representing the locations that a person has visited in a month, instead of the individual locations. We settled on this time interval since a shorter time span (such as a day) contains very little predictive information, while a longer span (one semester) groups together distinct time spans that may lead to divergent behaviors, such as exam periods. We create a sequence embedding by taking a weighted average of the location vectors included in the sequence, using the time spent at each location as weights, thus increasing the importance of locations at which the person spent more time.

### 5 Probing Location Representations

While some of the methods we use (i.e., skip-gram) have been used in the past to represent locations for certain tasks, there has been less work studying them intrinsically. We propose surface level tasks to probe the properties encoded in location embeddings, which are important to gain a deeper understanding of the type of information they capture. We split surface level tasks into two categories: those that focus on individual locations and those that focus on location sequences. In addition to these surface level tasks, we propose a set of downstream prediction tasks to validate the utility of such embeddings.

#### 5.1 Surface Level Location Tasks

With these tasks, we examine two properties that should be encoded in location representations: location functionality and physical proximity. To directly compare how well each method encodes these semantic properties, we propose a metric to measure each property. We are inspired by Ye and Skiena (2019), who use similar methods to analyze properties of name embeddings (representations of

people’s names). We borrow their method of analysis, measuring overlaps in the  $N$  nearest neighbors for various values of  $N$ , but they analyze a different property, namely the gender associated with the name.

**Functionality Overlap.** Each location in our dataset is annotated with its functionality, including two functionalities for mixed-use buildings, e.g., a class building that also contains labs. For each location, we calculate the percentage of its nearest neighbors in the vector space that share at least one functionality; a higher value indicates that the embeddings more distinctly capture functionality. We compute nearest neighbors using cosine similarity.

**Physical Distance.** We compute the distance in kilometers between a location and its nearest neighbors, and average the distances. This allows us to measure exactly how far a location is from its nearest neighbors; a lower number for this metric correlates with an increased physical proximity.

## 5.2 Surface Level Sequence Tasks

Our surface level sequence tasks are inspired by the methodology proposed by Conneau et al. (2018) to probe sentence embeddings. Many of those tasks focus on syntax, which is not relevant for our use case, but we adapt their task for location-presence and propose probing for functionality-presence.

**Location Presence.** We propose a binary location-presence classification task (LocPres). We create classifiers for each location, predicting if the location appears in a sequence. We average the results across all locations with at least one hundred positive and negative examples (resulting in being able to assess 83 locations out of 132).

**Functionality Presence.** We also propose a functionality-presence task (FuncPres). Given a sequence embedding, we predict if it includes locations of a certain functionality. We use a binary classification setup that mirrors the one used for the location-presence task. We treat the classification of either the primary or secondary functionalities assigned to locations as correct. As with the location-presence task, we average results over all functionalities with at least one hundred training instances from each class (accounting for 11 functionalities out of 13).

## 5.3 Downstream Application-Based Tasks

In addition to surface level tasks, we want to understand what other human-centric information is

encoded in location sequence embeddings. Our hypothesis is that the way in which students spend their time may be indicative of certain information about them; an example of students’ diverse behavior on campus is shown in Figure 3. Using the dataset described in Section 3.1, we propose seven classification tasks: five tasks with two classes (major depression, all depression, gender, sleep satisfaction, and GPA), one task with three classes (to predict which school a student is enrolled in, e.g. business or engineering), and one task with four classes (to predict class year).

Sleep satisfaction is reported in a survey (Section 3.1) on a five-point Likert scale; the top three responses are mapped to a positive class, and the bottom two to a negative class. As semester GPA is continuous, we formulate the binary classification as less than or greater than 3.5 (between A- and B+). Depression is measured using the standard PHQ-8 survey; using a clinically validated algorithm (Kroenke et al., 2001), we classify major depression (binary), along with major and other depression (a weaker diagnosis); we label the former as “major depression” and the latter as “all depression.” For the other tasks, we filter out underpopulated classes, going from 18 to three classes for school, from five to two for gender, and from five to four for class year. We use a classification approach over regression because we hope that this work can be used to identify at-risk students.

## 6 Experimental Setup

We perform 10-fold cross validation on 729 instances, where each instance represents a student. Preliminary classification experiments were conducted on a small subset using SVM with linear and RBF kernels, random forests, decision trees, and Naïve Bayes, yet linear SVM had the most robust performance. Accordingly, our experiments consist of classification tasks using linear SVM. As many of the classes are unbalanced, we more heavily weight updates for the minority class(es) by modifying the loss function to use a weight that is inversely proportional to the class’s prevalence.

To predict a student attribute, we create one vector for each month of data collection pertaining to each student, using the process described in Section 4.4. Our training framework is illustrated in Figure 4. We start by feeding the sequence vectors through a SVM classifier, which predicts month-level labels. These are then concatenated to form a

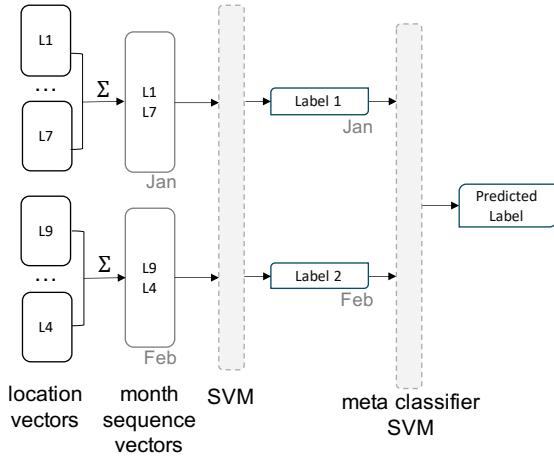


Figure 4: The framework for downstream prediction tasks.

student instance and are passed to a meta-classifier that decides the final class label for that student. We use the meta-classification approach to allow the first classifier more data to learn from; without this approach, the number of input samples is relatively small (729). The process for surface level sequence tasks is similar, but no meta-classifier is used, as the gold standard labels have a month-level granularity.

## 7 Results and Discussion

Figure 5 and Tables 3 and 5 show the results obtained for the probing tasks. In addition to the loc2vec trajectory and text-based models, we run our experiments with two combination models, using the methods discussed in Section 4.3. We employ the Reddit variation for these combination models due to its strong performance on downstream tasks; we incorporate one model using concatenation and a model using retrofitting. We refer to these models as “Loc2V-Reddit,” and “Reddit-Retrofit,” respectively.

We compare our classification performance against a random baseline. In order to introduce a stronger supervised baseline for our methods, we employ simpler location representations, in the form of one-hot vectors, which are passed as input in our supervised evaluation framework (Figure 4). We take the mean of those one-hot vectors to create month sequence vectors as we do for the embeddings.

### 7.1 Surface Level Location Tasks

For these tasks, we include an overall average baseline, where we compute the metric for all locations.

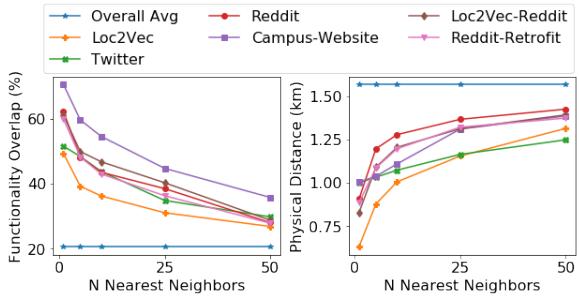


Figure 5: Results on surface level location tasks.

tions. The results, shown in Figure 5, lead to two unsurprising findings: text-based methods are better at encoding functionality, and the methods rooted in physical location are better at encoding distance. The results are somewhat skewed for the text-based representations such as “Campus-Website,” as some locations share a single page; however, this effect alone does not entirely explain the performance of that model on the functionality overlap task, as it is outperformed on the physical distance task.

One fascinating result is that the Twitter embeddings offer the best performance on the physical distance task by a method that does not utilize physical trajectories, which may be because this data is collected using geotags. People may tweet as they move between buildings, blurring the line between tweets in adjacent locations. We also observe that the methods that account for physical trajectories and text data can outperform those that use only text data; this is especially clear from the results for Loc2V-Reddit, which show stronger performance than Loc2Vec and Reddit individually for functionality overlap, and slightly stronger performance than Reddit for physical distance. This demonstrates one way in which we can create more robust representations of locations.

### 7.2 Surface Level Sequence Tasks

Overall, we note that all of our methods are easily able to surpass the random baseline. However, when it comes to the supervised one-hot vectorial representation, we see that traditional ways of representing text are able to best encode surface level information. This is because the sparse one-hot representation explicitly encodes information necessary for solving each task; location-presence is denoted by a value greater than one for the particular dimension, and functionality-presence is denoted by a value greater than one for various

	Loc Pres	Func Pres
Random Baseline	41.0	45.0
One-Hot Avg	<b>61.4</b>	<b>62.6</b>
Loc2V	54.8	55.6
Twitter	56.9	57.8
Reddit	56.9	58.3
Campus-Website	55.8	57.8
Loc2V-Reddit	57.9	59.7
Reddit-Retrofit	55.2	56.5

Table 3: Macro F1 scores (%) on surface level sequence tasks.

Task	# Cls	Inst	% in minority class
Class Year	4	721	22.33
Gender	2	714	49.44
School	3	522	9.77
Sleep	2	729	41.02
GPA	2	729	38.13
All Depression	2	729	18.93
Major Depression	2	729	11.66

Table 4: Class balance for downstream tasks. Instances are reported after filtering small classes.

dimensions.

We find that the text-based methods lead to stronger performance, as compared to their location-trajectory-based counterpart. This confirms that the superior encoding of functionality discussed in Section 7.1 is still discernible with aggregated sequence vectors.

Among all of our proposed methods, the concatenation of trajectory-embeddings and text-based embeddings (Loc2V-Reddit) leads to the strongest results on these tasks. The results on both tasks are completely unmatched by the other methods, indicating that the additional semantic information from concatenation leads to stronger representations.

### 7.3 Downstream Tasks

We evaluate our embedding methods on the seven downstream tasks introduced in Section 5.3: class year, gender, school enrollment, sleep satisfaction, GPA, all depression, and major depression. These tasks were designed to demonstrate the utility of various location representations in predicting a diverse set of attributes. The overall results for each model are listed in Table 5; we use macro F1 score as our metric. Table 4 shows the size of the minority class for each task. This imbalance and our

relatively small data size made it challenging to achieve strong results on some tasks, although we generally were able to improve upon the baselines. Across all the tasks, predicting depression has the most potential for real-world impact, but also showcases the most imbalanced data distribution. With more data, we believe that patterns could be learned in a more robust way.

For the task of school prediction, we greatly improve upon the random baseline even though the data is very imbalanced; this could be because this attribute is clearly linked to where people go on campus, as is class year. For example, freshmen typically live in dorms and eat in dining halls, while seniors often live off campus; computer science students attend classes in different places than English students. The strong performance on the gender prediction task may be explained by the real-world bias entailed in the school of enrollment; e.g., fewer women are enrolled in engineering, so they are less likely to visit engineering buildings. The strong performance on predicting class year with one-hot encodings can be directly linked to the surface level task improvement: freshmen are more likely to visit certain types of locations like dorms (functionality-presence); performance is best among freshmen.

Among text-based methods, we see that the Reddit embeddings enable the best performance on most downstream tasks. Reddit contains the most expressive language compared to the other venues, because its users are able to write at length without a strict character limit or other formalities imposed by media such as Twitter. Furthermore, from manually examining a sample of the posts, the community seems to primarily encompass current and former undergraduate students, therefore establishing a community that is above all else a place for students to share and discuss their daily lives. Meanwhile, the tweets that we link to locations may encompass musings from faculty or visiting scholars, and brief statements that are unrelated to campus life. The campus website data is the furthest from the student experience, as it is devoid of any dynamic content, written in the dry format of informational style. As a result, it seems intuitive that Reddit, in addition to providing *definitional* information about locations (e.g., there are many posts comparing and discussing dormitories), also provides student’s *emotional* perspectives on them. We hypothesize that this closeness to student thoughts and feelings is what yields bet-

	Class	Year	Gender	School	Sleep	GPA	All	Depression Major
Random Baseline	25.0	50.0	30.0	50.0	49.0	45.0	41.0	
One-Hot Avg	<b>52.1</b>	56.8	61.8	49.4	51.8	48.2	46.6	
Loc2V	50.8	<b>61.0</b>	62.0	52.9	51.9	49.6	43.6	
Twitter	49.4	57.4	65.4	49.3	51.9	48.5	44.8	
Reddit	50.2	59.8	<b>66.3</b>	52.7	49.1	50.5	<b>47.7</b>	
Campus-Website	48.8	58.1	60.1	46.4	51.9	49.4	42.9	
Loc2V-Reddit	50.3	59.4	64.5	<b>53.7</b>	<b>52.7</b>	<b>50.8</b>	44.7	
Reddit-Retrofit	50.2	60.8	66.0	52.6	47.7	48.7	39.6	

Table 5: Macro F1 scores (%) on downstream tasks.

ter performance when predicting student attributes, compared to the other text-based methods.

Overall, while results vary between different tasks, we find that a method that accounts for both physical location trajectories and text data describing locations (Loc2V-Reddit) has a strong overall performance. Notably, it is the best performing model on three tasks and achieves large improvements over the supervised baseline on two additional tasks. Such a model should be considered in future work on location embeddings because of its robustness on varied tasks.

## 8 Conclusions

In this paper, we addressed the task of building and probing location embeddings. We investigated several strategies to construct them, as well as a suite of probing tasks to understand the type of information encoded within. First, we showed that while all embedding methods encode both physical distance and functionality, methods using trajectories yield better spatial representations and methods using text data better encode location functionality. We showed that, like in the case of sentence embeddings from natural language, sequence embeddings of location data are able to encode surface level information (location-presence, and functionality-presence), as well as information that can be effectively used in downstream tasks. Overall, we found that an embedding model that accounts for both location trajectories and text related to locations (Loc2V-Reddit) gives the best performance over a diverse range of downstream tasks, from prediction of depression or sleep to prediction of academic area of study.

Importantly, we also found that embeddings of locations tend to underperform more traditional one-hot encodings on surface-level tasks,

yet they generally outperform these representations on downstream tasks. This suggests that while such embeddings do not explicitly record distinct locations that people visit (thus being more privacy preserving and counteracting negative actions like stalking), they may be more effective for downstream applications that can yield positive outcomes, such as population-level mental health tracking or opt-in tracking for individuals who are in therapy.

Our code is publicly available at <http://lit.eecs.umich.edu/downloads.html>.

## Acknowledgment

We are grateful to Shiyu Qu, Jiaxin Ye, and Xinyi Zheng for assisting us with this study. This material is based in part upon work supported by the Precision Health initiative at the University of Michigan, by the Michigan Institute for Data Science (MIDAS), by the National Science Foundation (grant #1815291), and by the John Templeton Foundation (grant #61156). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the Precision Health initiative, MIDAS, the National Science Foundation, or John Templeton Foundation.

## References

- Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2017. [Fine-grained analysis of sentence embeddings using auxiliary prediction tasks](#). In *International Conference on Learning Representations*.

- Buru Chang, Yonggyu Park, Donghyeon Park, Seongsun Kim, and Jaewoo Kang. 2018. **Content-aware hierarchical point-of-interest embedding model for successive poi recommendation**. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 3301–3307. International Joint Conferences on Artificial Intelligence Organization.
- Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. **What you can cram into a single \$&#!#\* vector: Probing sentence embeddings for linguistic properties**. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136, Melbourne, Australia. Association for Computational Linguistics.
- Alessandro Crivellari and Euro Beinat. 2019. **From motion activity to geo-embeddings: Generating and exploring vector representations of locations, traces and visitors through large-scale mobility data**. *ISPRS International Journal of Geo-Information*, 8(3):134.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. **BERT: Pre-training of deep bidirectional transformers for language understanding**. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. 2015. **Retrofitting word vectors to semantic lexicons**. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 1606–1615*, Denver, Colorado. Association for Computational Linguistics.
- Shanshan Feng, Gao Cong, Bo An, and Yeow Meng Chee. 2017. **POI2Vec: Geographical latent representation for predicting future visitors**. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 102–108.
- Kurt Kroenke, Robert L Spitzer, and Janet BW Williams. 2001. **The PHQ-9: validity of a brief depression severity measure**. *Journal of General Internal Medicine*, 16(9):606–613.
- Xin Liu, Yong Liu, and Xiaoli Li. 2016. **Exploring the context of locations for personalized location recommendations**. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI’16*, pages 1188–1194. AAAI Press.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. **Distributed representations of words and phrases and their compositionality**. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.
- Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. **GloVe: Global vectors for word representation**. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.
- Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. **Deep contextualized word representations**. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.
- Gerard Salton and Christopher Buckley. 1988. **Term-weighting approaches in automatic text retrieval**. *Information processing & management*, 24(5):513–523.
- Xing Shi, Inkit Padhi, and Kevin Knight. 2016. **Does string-based neural MT learn source syntax?** In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1526–1534, Austin, Texas. Association for Computational Linguistics.
- Adir Solomon, Ariel Bar, Chen Yanai, Bracha Shapira, and Lior Rokach. 2018. **Predict demographic information using word2vec on spatial trajectories**. In *Proceedings of the 26th Conference on User Modeling, Adaptation and Personalization, UMAP ’18*, page 331339, New York, NY, USA. Association for Computing Machinery.
- Junting Ye and Steven Skiena. 2019. **The secret lives of names?: Name embeddings from social media**. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD ’19*, pages 3000–3008, New York, NY, USA. Association for Computing Machinery.
- M. Zhu, W. Chen, J. Xia, Y. Ma, Y. Zhang, Y. Luo, Z. Huang, and L. Liu. 2019. **Location2vec: A situation-aware representation for visual exploration of urban locations**. *IEEE Transactions on Intelligent Transportation Systems*, 20(10):3981–3990.

# Self-Supervised Learning for Pairwise Data Refinement

Gustavo Hernández Ábrego, Bowen Liang, Wei Wang, Zarana Parekh,  
Yinfei Yang, Yunhsuan Sung

Google Research, Mountain View, CA, USA

{gustavoha, bowenl, wangwe, zarana, yinfeiy, yhsung}@google.com

## Abstract

Pairwise data automatically constructed from weakly supervised signals has been widely used for training deep learning models. Pairwise datasets such as parallel texts can have uneven quality levels overall, but usually contain data subsets that are more useful as learning examples. We present two methods to refine data that are aimed at obtaining that kind of subsets in a self-supervised way. Our methods are based on iteratively training dual-encoder models to compute similarity scores. We evaluate our methods on de-noising parallel texts and training neural machine translation models. We find that: (i) The self-supervised refinement achieves most machine translation gains in the first iteration, but following iterations further improve its intrinsic evaluation. (ii) Machine translations can improve the de-noising performance when combined with selection steps. (iii) Our methods are able to reach the performance of a supervised method. Being entirely self-supervised, our methods are well-suited to handle pairwise data without the need of prior knowledge or human annotations.

## 1 Introduction

Deep learning models are widely adopted and have demonstrated their usefulness in many areas and applications. Despite their diversity, one common characteristic of these models is the large number of parameters that need to be adjusted during training (some recent models that have billions of parameters include T5 (Raffel et al., 2019) and GPT-2 (Radford et al., 2019)). This leads to the need of collecting large amounts of training examples. Pairwise data, that captures the relationship in two modalities, is used to train deep learning models such as Neural Machine Translation (NMT) (Wu et al., 2016), Question Answering (Wang et al., 2007), Image Captioning (Sharma et al., 2018), etc.

To train this kind of models, large-scale data can often be obtained from weak signals like text co-occurrence (Yang et al., 2018) or dictionary n-gram matching (Uszkoreit et al., 2010). For example, in the machine translation community, the large amount of multilingual text available on the internet has naturally led to the idea of using internet data to train NMT models (Resnik, 1999). This approach has proven advantageous but it has the drawback that data mined this way is intrinsically noisy (Resnik and Smith, 2003). Despite the poor quality, usually this kind of data contains a helpful subset that can be recovered through a process of data cleaning or refinement. Data cleaning could be implemented with linguistic knowledge such as its script, vocabulary, syntax, etc. Alternatively, a model can be trained on “clean” or “trusted” pairs that are verified through manual annotation. Both options can be highly effective, but the former is limited in scope and error-prone, while the latter can be costly due to the number of required annotated examples.

In this paper we introduce two self-supervised methods to obtain data subsets from noisy pairwise data that can be helpful to train dual-encoder (D-E) and neural machine translation (NMT) models. As noisy pairwise data, in our experiments we use parallel texts mined from the internet. Our methods do not require external knowledge (e.g. syntactic rules), language-dependent heuristics (e.g. script verification) or synthetic positive or negative training examples. By eliminating the need of annotations, our methods directly address the data labelling bottleneck. Our methods employ D-E models (Gillick et al., 2018) to learn a shared embedded space from the co-located text in the sentence pairs mined from the internet. Following Chidambaram et al. (2018) we use the embedding distance in the learned space as a measure of cross-lingual similarity between sentences. Our hypothesis is that

if higher scores are associated with cross-lingual similarity, pairs with higher scores will be closer to be actual translations of each other and, in that case, may be part of the data subset useful to train the models.

In our experiments, our methods show effective refining parallel texts mined from the internet. Much of the gains in the downstream evaluation are achieved in the first iteration of the method, but later iterations keep improving the D-E models. Despite being self-supervised, our methods show competitive performance when compared against a de-noising method that uses supervision.

## 2 Related Work

One line of the research that directly relates to our work is corpus filtering for training NMT models. Below we classify the related work into two categories depending on the amount of supervision needed (e.g. high quality parallel texts).

**(Semi-)Supervised Methods** Some data denoising methods simply use filtering rules or heuristics such as language identification of both the source and target texts, vocabulary checks, language model (syntactic) verification, and so on. In contrast to rule-based approaches, approaches like [Chen and Huang \(2016\)](#) and [Wang et al. \(2018c\)](#) train classifiers to distinguish in-domain vs. out-of-domain (or clean vs. noisy) data with a small parallel corpus, while other approaches build reference models on larger amounts of high-quality data ([Junczys-Dowmunt, 2018](#); [Defauw et al., 2019](#)). There are approaches that combine rules and heuristics with probabilistic models to determine the amount of noise in each sentence pair. In some cases these systems are designed as targeted efforts to denoise a particular dataset. Bicleaner ([Sánchez-Cartagena et al., 2018](#)), in relationship to the ParaCrawl ([Esplà et al., 2019](#)) data, is an example of that approach.

**Unsupervised Methods** In contrast to the supervised methods, unsupervised methods do not require good-quality data to be available. Recent work ([Zhang et al., 2020](#)) leverages pre-trained language models and synthetic data ([Vyas et al., 2018](#)), in place of true supervision. Some efforts focus on using monolingual corpora and align them through bootstrapping in order to generate sentence pairs ([Tran et al., 2020](#); [Ruiter et al., 2020](#)), while others train a model with noisy data directly to gen-

erate embeddings and score the data ([Chaudhary et al., 2019](#)). [Wang et al. \(2018b\)](#) use two NMT models taken from two training epochs to decide which data to use in order to improve the training efficiency and to show a de-noising effect. Our methods here try to take advantages of all of these approaches. [Koehn et al. \(2018\)](#) and [Koehn et al. \(2019\)](#) summarize findings of the WMT corpus filtering efforts, though our work here primarily examines a self-supervised method in the context of de-noising, rather than on a targeted filtering effort.

Our methods are unsupervised. We use dual-encoder models, rather than an encoder-decoder architecture, to model pairwise data and let the model self-supervise itself or, further, be co-trained with an NMT model to refine the training data.

## 3 Dual-Encoder Model

Dual-encoder (D-E) models have demonstrated to be an effective learning framework applied to both supervised ([Henderson et al., 2017](#); [Gillick et al., 2019](#)) and unsupervised tasks ([Cer et al., 2018](#); [Chidambaram et al., 2018](#)). A multi-task D-E model consists of two encoders and a combination function for each of the tasks. In the context of the D-E framework, the selection of bilingual text can be interpreted as a ranking problem where, with  $y_i$  as the true target of source sentence  $x_i$ ,  $P(y_i|x_i)$  is ranked above all the other target candidates in  $\mathcal{Y}$ .  $P(y_i|x_i)$  can be expressed as a log-linear model but, for practical reasons, we approximate the full set of target candidates  $\mathcal{Y}$  with a sample ([Henderson et al., 2017](#)). When training in a batch,  $P(y_i|x_i)$  can be approximated as:

$$P(y_i|x_i) \approx \frac{e^{\phi(x_i, y_i)}}{e^{\phi(x_i, y_i)} + \sum_{n=1, n \neq i}^N e^{\phi(x_i, y_n)}} \quad (1)$$

where  $N$  is the size of a batch and  $\phi$  is a similarity function. In such a way, model training can be done by optimizing a log-likelihood loss function:

$$\mathcal{L} = -\frac{1}{N} \sum_{i=1}^N \log \frac{e^{\phi(x_i, y_i)}}{e^{\phi(x_i, y_i)} + \sum_{n=1, n \neq i}^N e^{\phi(x_i, y_n)}} \quad (2)$$

Based on the results of [Yang et al. \(2019a\)](#) with additive margin softmax ([Wang et al., 2018a](#)), we modify our loss function to include margin  $m$ :

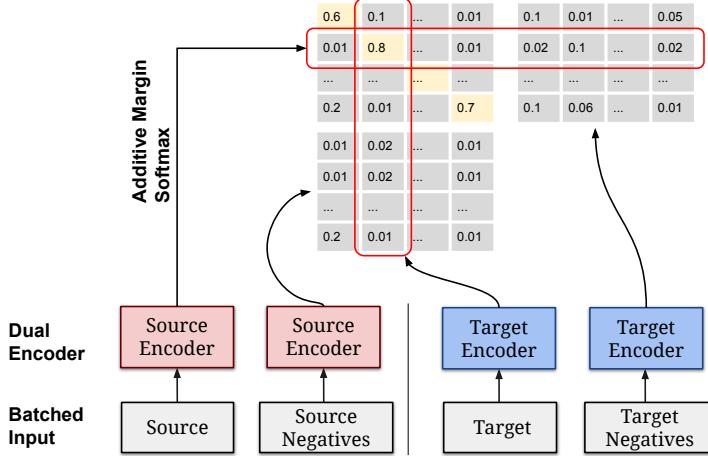


Figure 1: D-E model training with hard negatives. The encoders with the same color share parameters. The dot product scoring function makes it easy to compute pairwise scores by doing matrix multiplications. The highlighted diagonal indicates the dot products of the source and target texts. The additive margin softmax is applied at every row (source→target) and column (target→source).

$$\mathcal{L}_{ams} = -\frac{1}{N} \sum_{i=1}^N \log \frac{e^{\phi(x_i, y_i) - m}}{e^{\phi(x_i, y_i) - m} + \sum_{n=1, n \neq i}^N e^{\phi(x_i, y_n)}} \quad (3)$$

When using the dot product as similarity function  $\phi$ , a single matrix multiplication can be used to efficiently compute scores for all the examples in the batch. When set to learn from clean cross-lingual paired texts, a D-E model can be used to learn strong cross-lingual embeddings for bitext retrieval as shown in Guo et al. (2018) and Yang et al. (2019a). The challenge is to learn similar embeddings when training D-E models on noisy data.

### 3.1 Model Configuration

In our experiments we use D-E models with hard negatives sampling (Guo et al., 2018). Similar to Yang et al. (2019a), our models are trained bidirectional so the rankings in both directions, source to target and target to source, are optimized. But in contrast to Yang et al. (2019a) we do not share the parameters between the source and target encoders. In our initial experiments training NMT models we found that, under noisy conditions, there is improvement of close to 1 BLEU point when using D-E models that use specific encoders for each language. Figure 1 illustrates our training approach. For our encoders we use 3-layer transformers (Vaswani et al., 2017) in the encoders with hidden layers of size 512 and 8 attention heads.

We build vocabularies for each language separately. Given the noise in the data, the vocabularies

might not include all words in the source or target languages. We control the prevalence of words in the expected language with the vocabulary size. Our reasoning is that large vocabularies are more likely to include words in languages other than the expected. 200k most frequent words are used and 200k extra buckets are reserved for the out-of-vocabulary words found in the training. We use character- and word-level features to model the source and target inputs. For character-level representations, we decompose each word into all character n-grams within a range. For word-level representation, we sum the embeddings for its character n-grams and its word embedding. The final sentence representation is the output of the transformer layers as a 500-dimensional vector. We train the D-E models using SGD for 40M steps with a learning rate of 0.001. A fixed value of margin 0.2 is used in equation 3.

## 4 Our Approach: Self-Supervised Learning for Data Refinement

### 4.1 Training with Hard Negatives

As described in equation 1, and illustrated in figure 1, for every source sentence we use all target sentences, except its own, as negatives in a batch. We also augment the batch with hard negatives to improve the contrast between true translations and any other random sentence pairing. We mine the hard negatives using a separate D-E model to retrieve, for every sentence, the top  $N$  candidates that are not its counterpart in the pair. It is important

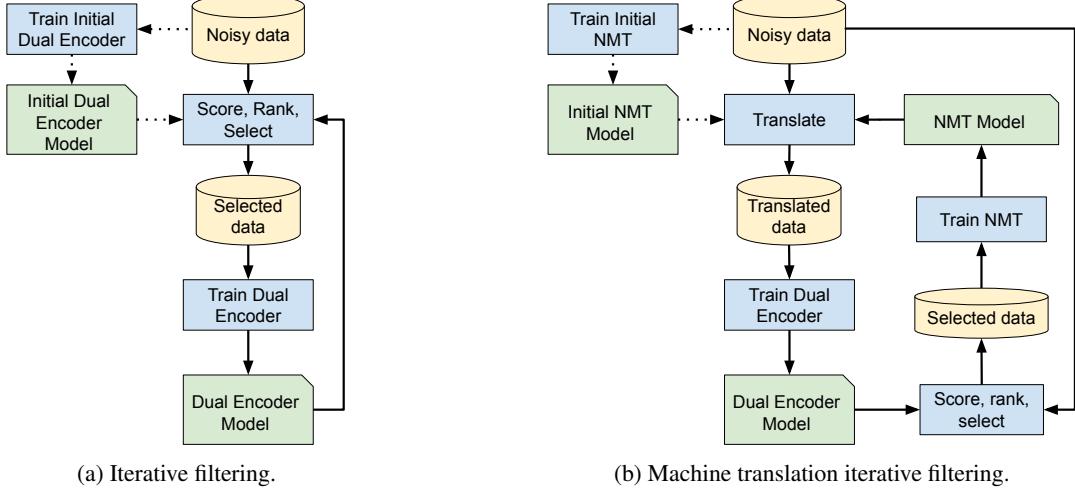


Figure 2: **[Iterative filtering (IF)]**: the scores of the dual-encoder are used to select the training material for the next model. **[Machine translation iterative filtering (MT-IF)]**: The D-E model is used to score the forward-translations from the NMT model, only the top-ranking sentence pairs are used to train the next NMT model.

to notice that the hard negatives in our method are retrieved, not generated or synthesized. We mine the hard negatives offline from the sentences in the ParaCrawl v1.0 data, or from the translations only when using translations as target sentences. Our negative-mining D-E has DNN layers, instead of transformer ones, with a reduced embedding size (25-dimensional). We mine hard negatives for both the source and target sentences. As shown in figure 1, the hard negatives are specific to each one of the sentence pairs but, when added to the batch, we use them as additional random negatives for all the other source sentences in the batch. We use a batch size of 128 examples and 5 hard-negatives per example. We augment the batch row-wise with hard negatives mined for the target sentences, and column-wise with hard negatives for the source.

In our self-supervised approach, we train D-E models with one dataset and use the models that we train to score the same data. Our hypothesis is that the scores are useful to rank the data in a way that makes it easy to filter out the noise. It is natural to believe that, in principle, a data-model cycle like this may not lead to much improvement because the trained models tend memorize the training data, including the noise. We break this cycle by adding a selection step to the process and avoiding to train the models with the same examples all the time. We propose a self-supervised method for pairwise data refinement based on data “iterative filtering” (*IF*). With this method we refine data that we use to train NMT models. By including the downstream task in our method, we formulate a second method as

an extension of the first one. We regard this second method as “machine translation-iterative filtering” (*MT-IF*). Both methods are illustrated in figure 2.

## 4.2 Iterative Filtering

We use the dot product between source and target embeddings as proxy of cross-lingual similarity. Once we score and select data to train one model, we can use that model to score and select data for the next one in an iterative way. The details of this method are shown in figure 2a and explained in algorithm 1.

We bootstrap this method by training an initial D-E model with all the pre-filtered data. It is important to notice that in each iteration we train the D-E model with a subset of the data (the selected data), but we score the entire set. This allows the method to recover useful data that may have been discarded in earlier iterations.

---

### Algorithm 1 Iterative filtering

---

- 1:  $\tau \leftarrow$  selection threshold
  - 2:  $D-E = \text{TrainDualEncoder}(\text{data})$
  - 3: **while**  $D-E$  improves **do**
  - 4:     scored data = Score(data;  $D-E$ )
  - 5:     ranked data = Rank(scored data)
  - 6:     selected data = Select(ranked data;  $\tau$ )
  - 7:      $D-E = \text{TrainDualEncoder}(\text{selected data})$
  - 8: **end while**
-

### 4.3 Machine Translation Iterative Filtering

In this method, the D-E model selects data to train an NMT model, rather than to train another D-E model. The NMT model then produces translations to train the D-E model. This way, the D-E and NMT models boost each other in a “co-training” way. The key to this method is to use the NMT model to generate the training data for the D-E model in order to improve its de-noising capabilities. Algorithm 2 explains this idea and figure 2b illustrates it.

As before, in every iteration the whole dataset is scored and ranked so sentence pairs that ranked low early on can be recovered in later iterations. In principle, forward-translation does not seem to be a good way to generate training data. One can anticipate that the models are prone to mimic the training data, including the noise. Just as in our first method, we break the cycle by adding a selection step based on the D-E scores and using only the top-ranking data to train the next NMT model.

## 5 Experimental Setup

**Machine Translation Model** To assess if we can recover useful subsets from noisy data, we train Transformer-Big (Vaswani et al., 2017) NMT models using data refined with our methods. To train the models, we split the source and target texts into pieces using bilingual sentence piece models (Kudo and Richardson, 2018) that were trained with the ParaCrawl v1.0 data only. We train for a maximum of 200k steps using (Shazeer and Stern, 2018) and pick the best checkpoint according to the performance on a validation set. The models are trained on Google’s Cloud TPU v3 with batch size 3072. In all our experiments, the configuration of the NMT models is kept the same with the only difference being the training data.

---

#### Algorithm 2 Machine translation iterative filtering

---

```

1:  $\tau \leftarrow$  selection threshold
2: NMT = TrainNMT(data)
3: while D-E improves or NMT improves do
4:   translated data = Translate(data; NMT)
5:   D-E = TrainDualEncoder(translated data)
6:   scored data = Score(data; D-E)
7:   ranked data = Rank(scored data)
8:   selected data = Select(ranked data;  $\tau$ )
9:   NMT = TrainNMT(selected data)
10: end while
```

---

	en-fr	en-de
All sentence pairs	4,235 M	4,591 M
Pre-filtered	289 M	282 M
70th percentile (for NMT)	87 M	85 M
80th percentile (for D-E)	58 M	56 M

Table 1: Number of sentence pairs in the ParaCrawl v1.0 data, and after prefiltering and selection.

**Data** In our experiments we use two language pairs: English to French (en-fr) and English to German (en-de). We use ParaCrawl v1.0 (Esplà et al., 2019) as training data. We apply light-weight pre-filtering steps to remove sentence pairs that: (i) are duplicated, (ii) have identical source and target texts, (iii) have empty sentences, or (iv) have a large difference in the number of tokens. For the last case, we compute the ratio of source over target tokens as:  $\rho = \frac{n_S + \alpha}{n_T + \alpha}$  with  $n_S$  and  $n_T$  being the number of tokens in the source and in the target respectively, and  $\alpha$  a token count tolerance. With an  $\alpha$  of 15, we discard a sentence pair if  $\rho$  is greater than 1.5. Similarly for the ratio of target over source tokens. We use WMT newstest 2012-2013 (Bojar et al., 2014) as the development set and we evaluate on two sets: WMT newstest 2014 and news discussion test 2015 for en-fr; WMT newstest 2014 and 2015 for en-de.

**Evaluation** As described in section 3, we trained the D-E models as rankers. Thus, we use the BUCC 2018 mining task (Zweigenbaum et al., 2018) as an intrinsic metric for the model. The task data consists of corpora for four language pairs including fr-en and de-en. For each language pair, the shared task provides a monolingual corpus for each language and a ground truth list containing true translation pairs. The task is to construct a list of translation pairs from the monolingual corpora, and evaluate them in terms of the F1 compared to the ground truth.

To test the end-performance of the NMT models in terms BLEU scores, we compute the detokenized and case-sensitive BLEU scores against the original references using an in-house reimplementation of the mteval-v14.pl script .

**Iterative Selection** In our experiments we ran 3 iterations of the *IF* method and 3 iterations of the *MT-IF* one.

To define the value of the selection thresholds, we conducted initial experiments to explore the

impact of the threshold when selecting the data to train the D-E models. Figure 3 shows the BUCC results, in terms of the best F1 measure and the area under the precision-recall curve (AUCPR), for D-E models trained with data selected using different thresholds. Even though there is not a single threshold that works best for both languages, models trained with data selected from the 70th or 80th percentiles produce the best results. Using either very low (below 0.2) or very high thresholds (above 0.95) leads to D-E models with lower results. We set the selection thresholds for the data to train the D-E models and to train the NMT models separately. For the former we use data on the 80th percentile, and on the 70th percentile for the latter. Our intuition was that we can be more stringent when selecting data to train the D-E because only high-ranking examples may be true translations to learn from. Table 1 shows the number of sentences in the ParaCrawl v1.0 en-fr and en-de datasets and the amount of sentences that the pre-filtering and selection steps, at the different thresholds, let through. The large number of sentence pairs that are eliminated via pre-filtering give an indication of how much noise there is in the data. It is worth noticing that the subset of data that we deem “useful” is two orders of magnitude smaller than the original data.

## 6 Results

### 6.1 Intrinsic Dual-Encoder Evaluation

Table 2 shows the BUCC mining task results for the D-E models trained with our methods in terms of F1

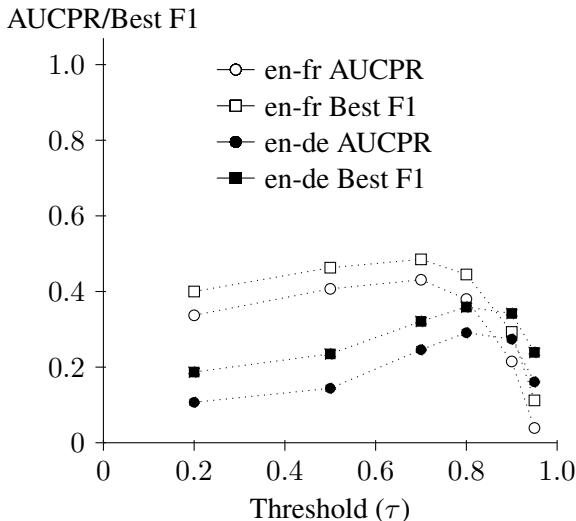


Figure 3: BUCC mining results of dual encoder models trained with data selected at different thresholds.

and AUCPR. As baseline we include the results of a D-E model trained with all the ParaCrawl v1.0 data after pre-filtering. The baseline performs poorly in both en-fr and en-de. The D-E models trained with the *IF* data produce good mining results starting from the very initial models, i.e. when using D-E models trained using hard negatives but no selection yet. The significant gains of  $IF_0$  over the baseline confirm our observations about the positive impact of hard negatives in cross-lingual tasks (Guo et al., 2018). In subsequent iterations (indices 1 to 3 in table 2) selection is used and the D-E models show steady improvement. The improvement in the AUCPR and F1 of the D-E models trained with the *MT-IF* data is quite remarkable. The performance for models trained with data from the first iteration of this method surpass the performance of models trained with the third iteration of the *IF* data and keep improving, but seem to plateau around the second iteration. For reference, we include in table 2 the AUCPR and F1 from embeddings generated with the public “universal-sentence-encoder-multilingual-large” v2 (Yang et al., 2019b) from TFHub<sup>1</sup> to show the performance of a D-E model trained on multiple large and non-public industry datasets. As expected, training on this kind of data is far better than de-noising, but the evaluation shows that our methods do a good job refining data, especially considering how much noise there is in the ParaCrawl datasets to start with.

### 6.2 Translation Evaluation

To illustrate the end-performance of our methods, table 3 shows the BLEU scores (Papineni et al., 2002) of NMT models trained with data subsets selected with our methods. The D-E models used to score the data in each iteration correspond to the same models reported in table 2. As baseline we use an NMT model trained with all the sentence pairs just after pre-filtering, i.e. selection is not used yet. For both our methods the NMT models show considerable improvement over the baseline. It is interesting that the initial NMT ( $IF_0$  in table 3), shows good improvement in spite of using a D-E whose only difference over baseline is the use of hard negatives. There is also noticeable improvement between the  $IF_0$  and  $IF_1$  results pointing to the fact that our process of scoring, ranking and selection is also useful to improve the

<sup>1</sup><https://tfhub.dev/google/universal-sentence-encoder-multilingual-large/2>

Method	en-fr		en-de	
	AUCPR	Best F1	AUCPR	Best F1
Pre-filtered data (baseline)	0.068	0.149	0.020	0.069
$IF_0$	0.246	0.330	0.094	0.179
$IF_1$	0.380	0.445	0.291	0.359
$IF_2$	0.570	0.600	0.372	0.415
$IF_3$	0.622	0.642	0.390	0.432
$MT\text{-}IF_1$	0.641	0.673	0.545	0.566
$MT\text{-}IF_2$	0.664	0.697	<b>0.600</b>	<b>0.620</b>
$MT\text{-}IF_3$	<b>0.676</b>	<b>0.707</b>	0.593	0.608
USE multi-lingual	0.824	0.812	0.861	0.815

Table 2: BUCC mining results of the dual-encoder models. The index in each experiment denotes the iteration. The USE multi-lingual model was trained using non-public industry datasets.

Method	en-fr		en-de	
	newstest2014	newsdiscusstest2015	newstest2014	newstest2015
Pre-filtered data (baseline)	0.303	0.297	0.196	0.239
$IF_0$	0.324	0.315	0.237	0.276
$IF_1$	0.342	0.343	<b>0.239</b>	0.281
$IF_2$	0.340	<b>0.352</b>	0.237	0.279
$IF_3$	0.342	0.348	0.236	0.283
Forward-translated data	0.305	0.306	0.203	0.243
$MT\text{-}IF_1$	0.342	0.346	0.237	0.280
$MT\text{-}IF_2$	0.343	0.348	0.235	0.283
$MT\text{-}IF_3$	<b>0.346</b>	0.349	0.236	<b>0.286</b>

Table 3: BLEU scores of the trained NMT models and the baseline models. The index in each experiment denotes the iteration.

NMT models. The second half of table 3 shows the BLEU scores when the NMT models are added to the refinement process in the *MT-IF* method. For a better reference, we train an NMT model with forward-translated sentence pairs using the baseline NMT model. Crucially, there is no selection on the forward-translated data to train this model. This NMT model does not show improvement relative to the baseline NMT model and confirms that distilling new training examples from forward translations provides little or no gain. In contrast to the BUCC evaluation from table 2, the downstream task does not seem to require several iterations to show good results. The BLEU scores of later iterations in the process only improve marginally as opposed to the steady improvement observed in the BUCC task.

### 6.3 Supervised vs Self-Supervised

We use Bicleaner to compare our methods against a supervised approach on the task of de-noising the ParaCrawl data, with the important caveat that

	en-fr	en-de
70th percentile after lang ID	29 M	27 M
Bicleaner v1.2	25 M	17 M

Table 4: Number of sentence pairs of selected data after language identification and in Bicleaner v1.2.

Bicleaner is not only supervised but tailored to de-noise this data. In that sense, our method would be in disadvantage especially because our D-E models were not trained with any signal related to the identity of the language. To add this missing component to our method, we use language identification as a post-processing step on the refined data. We use a pre-trained language identification method from Zhang et al. (2018) to filter out pairs where the source or target texts do not match the expected language. As around 30% of the training data gets discarded (table 1 vs table 4), the scores of the remaining data need to be re-ranked in preparation for the selection step. We train new NMT models using only the sentence pairs that get ranked in the

70th percentile and filtered by the language identification. We compare the models against similar NMT models trained with the Bicleaner v1.2 data downloaded from the ParaCrawl website<sup>2</sup>. Table 4 shows the number of sentence pairs used to train the NMT models after applying language identification and in the Bicleaner v1.2 data.

To isolate the effects of language identification, we compare NMT models trained with data from our methods against similar models trained with data that went through language identification also but, as in previous baselines, no selection was used.

As shown in Table 5, using language identification on the training data boosts the performance. The NMT models trained with the data refined with our methods still show considerable improvement over not using selection, making evident that there is still much room for data refinement after language identification. Our method shows very competitive results against the NMT models trained using the Bicleaner v1.2 data, surpassing the BLEU scores in en-fr and getting very similar performance in en-de. It is interesting that, with the addition of language identification, our self-supervised method can remove noise just as effectively as a targeted effort to denoise the ParaCrawl data.

#### 6.4 Iterative Data Refinement

To verify the effectiveness of our methods in finding useful subsets contained in the noisy data, we analyze the results of our models when scoring true sentence pairs versus scoring pairs that are not actual translations. For this analysis, we leverage the BUCC mining task and compute the dot products of “ground truth” pairs using our D-E models. Figure 4 shows box plots of the dot products for both en-fr and en-de BUCC data. For reference, we compute the dot products of the “nearest negative” of each source sentence. We reuse the retrieval results from the D-E intrinsic evaluation (subsection 6.1) to define the nearest negative as the target sentence with the highest dot product that is not its actual translation. This leads to 9,086 ground truth and nearest negative dot products for en-fr and 9,580 for en-de whose distributions are displayed in the box plots in figure 4. Starting with the baseline D-E models, the dot products of the ground truth and the nearest negative are very close in value. This is evident by the fact that their difference (also plotted in figure 4) is very close to 0. The differ-

ence starts to grow with the  $IF_0$  models, showing that hard-negatives are useful to increase the separation between the dot products of both classes. For the  $IF$  method, the difference between ground truth and nearest negative keeps growing steadily with every iteration. This confirms the progression observed in the AUCPR and F1 measures in table 2. For the  $MT\text{-}IF$  models, the score difference between ground truth and nearest negative is already significant in the first iteration, but it does not progress much further in later iterations. This also confirms the observations for these models in the BUCC mining results from table 2. The fact that the dot products of our models show good levels of separation of each class corroborate, from the data analysis standpoint, that both our methods are effective in separating useful data samples from the noisy dataset.

#### 6.5 Discussion

**Intrinsic vs downstream evaluations** Our self-supervised methods seem to naturally improve the quality of the refined data, as measured by the results of the BUCC parallel text mining task. However, most of the BLEU score gains are achieved on the first iteration. One possible explanation is that the BUCC evaluation is a closer match to the ranking task used to train the D-E model. Another possibility is that, given that different sequences can produce the same BLEU scores, there may be improvements in the translation quality that the BLEU scores do not reflect. Making the method more aware of the downstream translation task and gaining insight into the translation quality are interesting lines of future work.

**Language identification impact** In noisy data, language identification seems to play a significant role. In our experiments we applied it as a post-process but we are interested in applying it as part of the pre-filtering process, or integrated as part of our scoring models in the future.

**Breaking the data-model memorization cycle** Training NMT models directly with translated data did not produce gains over the baseline. But we found significant gains when instead we used the translated data to train D-E models and used the models to score and select data to in turn train the NMT models. We see this as confirmation that it is possible to break the data-model memorization cycle by co-training models using different training goals.

<sup>2</sup><https://paracrawl.eu/v1>

Method	en-fr		en-de	
	newstest2014	newsdiscusstest2015	newstest2014	newstest2015
Pre-filtered data lang ID	0.336	0.346	0.239	0.279
Bicleaner v1.2 data	0.363	0.370	<b>0.274</b>	<b>0.316</b>
<i>IF</i> <sub>1</sub>	<b>0.369</b>	<b>0.373</b>	0.263	0.306
<i>IF</i> <sub>2</sub>	<b>0.369</b>	0.369	0.267	0.308
<i>IF</i> <sub>3</sub>	0.366	0.372	0.269	0.314
<i>MT-IF</i> <sub>1</sub>	0.361	0.365	0.263	0.308
<i>MT-IF</i> <sub>2</sub>	0.363	0.370	0.262	0.303
<i>MT-IF</i> <sub>3</sub>	0.360	0.364	0.259	0.303

Table 5: BLEU scores of the NMT models using language identification and compared against Bicleaner.

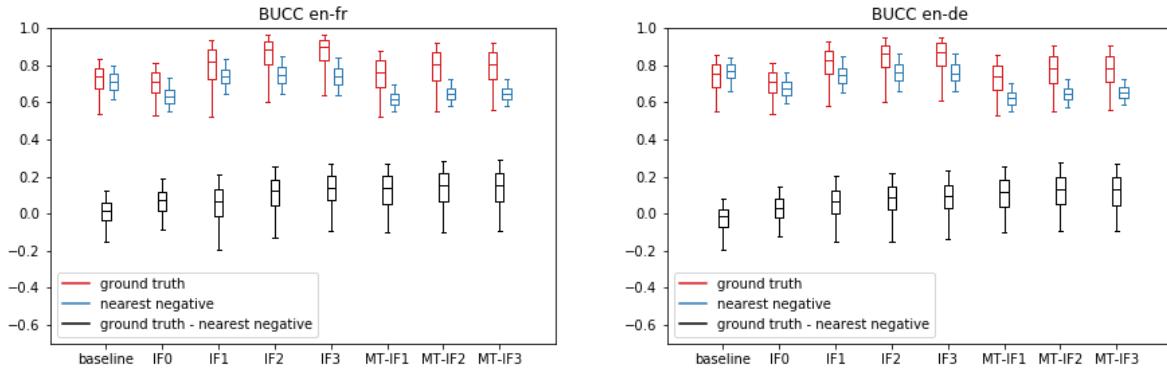


Figure 4: Dot product distributions for the ground truth and nearest negative from the BUCC mining task. The box plots represent the (5,25,50,75,95)-percentile of the dot product distribution for each method and iteration.

## 7 Conclusions

We introduced two self-supervised methods to refine pairwise data aimed at selecting useful subsets from noisy data. In our experiments we used parallel texts mined from the internet as example of the weakly constructed pairwise data to refine. Our methods do not require linguistic knowledge or human annotated data. They use iterative selection of the data to train two kinds of models. Our first method is based on self-boosting dual-encoder models iteratively. We applied this method to denoise data to train NMT models. Our second method integrates the NMT models into the iterative process to generate translations that, after a selection step, are used to train the dual-encoder models. Our results show that most of the gains in terms of BLEU score can be achieved in the first iteration of our methods, but later iterations keep improving the performance of the dual-encoder models in the BUCC evaluation. In our experiments, using translated text in combination with a selection step helped to improve the de-noising capabilities of the dual-encoder models. We observed that selection is effective to break the model-data

memorization cycle. One characteristic that our self-supervised methods do not seem to capture well is an indication of the language identity. If we use language identification on the denoised data as a post-processing step, the performance, in terms of BLEU scores, turns very competitive against supervised targeted efforts tailored to remove noise from the dataset. These results encourage us to pursue future lines of work that include using cross-attention in the pairwise data to better capture the relationship in the pairs. Also, specific to parallel sentences mined from the internet, we would like to explore ways to include language identification in the models. On the other hand, it seems natural to leverage the self-supervision characteristics of our methods and apply them to language pairs where noisy internet data may be available but annotated data is not. Lastly, we are interested in expanding our methods to other pairwise data such as text-image pairs.

## Acknowledgments

The authors would like to thank Noah Constant and the anonymous reviewers for their valuable feedback and suggestions to improve this paper.

## References

- Ondřej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, et al. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the ninth workshop on statistical machine translation*, pages 12–58.
- Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder for English. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 169–174, Brussels, Belgium. Association for Computational Linguistics.
- Vishrav Chaudhary, Yuqing Tang, Francisco Guzman, Holger Schwenk, and Philipp Koehn. 2019. Low-resource corpus filtering using multilingual sentence embeddings. *Proceedings of the Fourth Conference on Machine Translation (Volume 3: Shared Task Papers, Day 2)*.
- Boxing Chen and Fei Huang. 2016. Semi-supervised convolutional networks for translation adaptation with tiny amount of in-domain data. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning (CoNLL)*, pages 314–323.
- Muthuraman Chidambaran, Yinfei Yang, Daniel Cer, Steve Yuan, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Learning cross-lingual sentence representations via a multi-task dual-encoder model. *CoRR*, abs/1810.12836.
- A. Defauw, Sara Szoc, Anna Bardadym, Joris Brabers, Frederic Everaert, Roko Mijic, K. Scholte, Tom Vanallemeersch, Koen Van Winckel, and J. V. D. Bogaert. 2019. Misalignment detection for web-scraped corpora: A supervised regression approach. *Informatics*, 6:35.
- Miquel Esplà, Mikel Forcada, Gema Ramírez-Sánchez, and Hieu Hoang. 2019. ParaCrawl: Web-scale parallel corpora for the languages of the EU. In *Proceedings of Machine Translation Summit XVII Volume 2: Translator, Project and User Tracks*, pages 118–119, Dublin, Ireland. European Association for Machine Translation.
- Daniel Gillick, Sayali Kulkarni, Larry Lansing, Alessandro Presta, Jason Baldridge, Eugene Ie, and Diego Garcia-Olano. 2019. Learning dense representations for entity retrieval. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 528–537, Hong Kong, China. Association for Computational Linguistics.
- Daniel Gillick, Alessandro Presta, and Gaurav Singh Tomar. 2018. End-to-end retrieval in continuous space. *arXiv preprint arXiv:1811.08008*.
- Mandy Guo, Qinlan Shen, Yinfei Yang, Heming Ge, Daniel Cer, Gustavo Hernández Ábrego, Keith Stevens, Noah Constant, Yun-hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Effective parallel corpus mining using bilingual sentence embeddings. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 165–176. Association for Computational Linguistics.
- Matthew Henderson, Rami Al-Rfou, Brian Strope, Yun-Hsuan Sung, László Lukács, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. 2017. Efficient natural language response suggestion for smart reply. *CoRR*, abs/1705.00652.
- Marcin Junczys-Dowmunt. 2018. Dual conditional cross-entropy filtering of noisy parallel corpora. In *Proceedings of the Third Conference on Machine Translation, Volume 2: Shared Task Papers*, pages 901–908, Belgium, Brussels. Association for Computational Linguistics.
- Philipp Koehn, Francisco Guzman, Vishrav Chaudhary, and Juan Pino. 2019. Findings of the wmt 2019 shared task on parallel corpus filtering for low-resource conditions. In *Proceedings of the Fourth Conference on Machine Translation*, Florence, Italy. Association for Computational Linguistics.
- Philipp Koehn, Huda Khayrallah, Kenneth Heafield, and Mikel L. Forcada. 2018. Findings of the wmt 2018 shared task on parallel corpus filtering. In *Proceedings of the Third Conference on Machine Translation*, Belgium, Brussels. Association for Computational Linguistics.
- Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.
- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv e-prints*.
- Philip Resnik. 1999. Mining the web for bilingual text. In *Proceedings of the 37th Annual Meeting of the*

- Association for Computational Linguistics on Computational Linguistics, pages 527–534. Association for Computational Linguistics.
- Philip Resnik and Noah A Smith. 2003. The web as a parallel corpus. *Computational Linguistics*, 29(3):349–380.
- Dana Ruiter, Cristina España-Bonet, and Josef van Genabith. 2020. Self-induced curriculum learning in neural machine translation. *arXiv preprint arXiv:2004.03151*.
- Víctor M. Sánchez-Cartagena, Marta Bañón, Sergio Ortiz-Rojas, and Gema Ramírez-Sánchez. 2018. Prompsit’s submission to wmt 2018 parallel corpus filtering shared task. In *Proceedings of the Third Conference on Machine Translation, Volume 2: Shared Task Papers*, Brussels, Belgium. Association for Computational Linguistics.
- Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2556–2565, Melbourne, Australia. Association for Computational Linguistics.
- Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. *arXiv preprint arXiv:1804.04235*.
- Chau Tran, Yuqing Tang, Xian Li, and Jiatao Gu. 2020. Cross-lingual retrieval for iterative self-supervised training. *arXiv preprint arXiv:2006.09526*.
- Jakob Uszkoreit, Jay M. Ponte, Ashok C. Popat, and Moshe Dubiner. 2010. Large scale parallel document mining for machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics, COLING ’10*, pages 1101–1109, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.
- Yogarshi Vyas, Xing Niu, and Marine Carpuat. 2018. Identifying semantic divergences in parallel text without annotations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1503–1515, New Orleans, Louisiana. Association for Computational Linguistics.
- Feng Wang, Jian Cheng, Weiyang Liu, and Haijun Liu. 2018a. Additive margin softmax for face verification. *IEEE Signal Processing Letters*, 25(7):926–930.
- Mengqiu Wang, Noah A. Smith, and Teruko Mitamura. 2007. What is the Jeopardy model? a quasi-synchronous grammar for QA. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 22–32, Prague, Czech Republic. Association for Computational Linguistics.
- Rui Wang, Masao Utiyama, and Eiichiro Sumita. 2018b. Dynamic sentence sampling for efficient training of neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 298–304, Melbourne, Australia. Association for Computational Linguistics.
- Wei Wang, Taro Watanabe, Macduff Hughes, Tetsuji Nakagawa, and Ciprian Chelba. 2018c. Denoising neural machine translation training with trusted data and online data selection. In *Proceedings of the Third Conference on Machine Translation*, pages 133–143. Association for Computational Linguistics.
- Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google’s neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.
- Yinfei Yang, Gustavo Hernández Ábreo, Steve Yuan, Mandy Guo, Qinlan Shen, Daniel Cer, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2019a. Improving multilingual sentence embedding using bi-directional dual encoder with additive margin softmax. *CoRR*, abs/1902.08564.
- Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernández Ábreo, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2019b. Multilingual universal sentence encoder for semantic retrieval. *arXiv preprint arXiv:1907.04307*.
- Yinfei Yang, Steve Yuan, Daniel Cer, Sheng-yi Kong, Noah Constant, Petr Pilar, Heming Ge, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Learning semantic textual similarity from conversations. In *Proceedings of The Third Workshop on Representation Learning for NLP*, pages 164–174, Melbourne, Australia. Association for Computational Linguistics.

Boliang Zhang, Ajay Nagesh, and Kevin Knight. 2020.  
[Parallel corpus filtering via pre-trained language models](#). *arXiv:2005.06166*.

Yuan Zhang, Jason Riesa, Daniel Gillick, Anton Bakalov, Jason Baldwin, and David Weiss. 2018.  
[A fast, compact, accurate model for language identification of codemixed text](#). In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 328–337, Brussels, Belgium. Association for Computational Linguistics.

Pierre Zweigenbaum, Serge Sharoff, and Reinhard Rapp. 2018. Overview of the third bucc shared task: Spotting parallel sentences in comparable corpora. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Paris, France. European Language Resources Association (ELRA).

# A Survey of the State of Explainable AI for Natural Language Processing

Marina Danilevsky, Kun Qian, Ranit Aharonov, Yannis Katsis, Ban Kawas, Prithviraj Sen

IBM Research – Almadem

mdanile@us.ibm.com, {qian.kun, Ranit.Aharonov2}@ibm.com

yannis.katsis@ibm.com, {bkawas, senp}@us.ibm.com

## Abstract

Recent years have seen important advances in the quality of state-of-the-art models, but this has come at the expense of models becoming less interpretable. This survey presents an overview of the current state of Explainable AI (XAI), considered within the domain of Natural Language Processing (NLP). We discuss the main categorization of explanations, as well as the various ways explanations can be arrived at and visualized. We detail the operations and explainability techniques currently available for generating explanations for NLP model predictions, to serve as a resource for model developers in the community. Finally, we point out the current gaps and encourage directions for future work in this important research area.

## 1 Introduction

Traditionally, Natural Language Processing (NLP) systems have been mostly based on techniques that are inherently explainable. Examples of such approaches, often referred to as *white box* techniques, include rules, decision trees, hidden Markov models, logistic regressions, and others. Recent years, though, have brought the advent and popularity of *black box* techniques, such as deep learning models and the use of language embeddings as features. While these methods in many cases substantially advance model quality, they come at the expense of models becoming less interpretable. This obfuscation of the process by which a model arrives at its results can be problematic, as it may erode trust in the many AI systems humans interact with daily (e.g., chatbots, recommendation systems, information retrieval algorithms, and many others). In the broader AI community, this growing understanding of the importance of explainability has created an emerging field called Explainable AI (XAI). However, just as tasks in different fields are more amenable to particular approaches, explainability

must also be considered within the context of each discipline. We therefore focus this survey on XAI works in the domain of NLP, as represented in the main NLP conferences in the last seven years. This is, to the best of our knowledge, the first XAI survey focusing on the NLP domain.

As will become clear in this survey, explainability is in itself a term that requires an explanation. While explainability may generally serve many purposes (see, e.g., [Lertvittayakumjorn and Toni, 2019](#)), our focus is on explainability from the perspective of an end user whose goal is to understand how a model arrives at its result, also referred to as the *outcome explanation problem* ([Guidotti et al., 2018](#)). In this regard, explanations can help users of NLP-based AI systems build trust in these systems' predictions. Additionally, understanding the model's operation may also allow users to provide useful feedback, which in turn can help developers improve model quality ([Adadi and Berrada, 2018](#)).

Explanations of model predictions have previously been categorized in a fairly simple way that differentiates between (1) whether the explanation is for each prediction individually or the model's prediction process as a whole, and (2) determining whether generating the explanation requires post-processing or not (see [Section 3](#)). However, although rarely studied, there are many additional characterizations of explanations, the most important being the techniques used to either generate or visualize explanations. In this survey, we analyze the NLP literature with respect to both these dimensions and identify the most commonly used *explainability and visualization techniques*, in addition to *operations* used to generate explanations ([Sections 4.1–Section 4.3](#)). We briefly describe each technique and point to representative papers adopting it. Finally, we discuss the common *evaluation techniques* used to measure the quality of explanations ([Section 5](#)), and conclude with a discussion of gaps and challenges in developing success-

ful explainability approaches in the NLP domain (Section 6).

**Related Surveys:** Earlier surveys on XAI include Adadi and Berrada (2018) and Guidotti et al. (2018). While Adadi and Berrada provide a comprehensive review of basic terminology and fundamental concepts relevant to XAI in general, our goal is to survey more recent works in NLP in an effort to understand how these achieve XAI and how well they achieve it. Guidotti et al. adopt a four dimensional classification scheme to rate various approaches. Crucially, they differentiate between the “explanator” and the black-box model it explains. This makes most sense when a surrogate model is used to explain a black-box model. As we shall subsequently see, such a distinction applies less well to the majority of NLP works published in the past few years where the same neural network (NN) can be used not only to make predictions but also to derive explanations. In a series of tutorials, Lecue et al. (2020) discuss fairness and trust in machine learning (ML) that are clearly related to XAI but not the focus of this survey. Finally, we adapt some nomenclature from Arya et al. (2019) which presents a software toolkit that can help users lend explainability to their models and ML pipelines.

Our goal for this survey is to: (1) provide the reader with a better understanding of the state of XAI in NLP, (2) point developers interested in building explainable NLP models to currently available techniques, and (3) bring to the attention of the research community the gaps that exist; mainly a lack of formal definitions and evaluation for explainability. We have also built an interactive website providing interested readers with all relevant aspects for every paper covered in this survey.<sup>1</sup>

## 2 Methodology

We identified relevant papers (see Appendix A) and classified them based on the aspects defined in Sections 3 and 4. To ensure a consistent classification, each paper was individually analyzed by at least two reviewers, consulting additional reviewers in the case of disagreement. For simplicity of presentation, we label each paper with its main applicable category for each aspect, though some papers may span multiple categories (usually with varying degrees of emphasis.) All relevant aspects for every

<sup>1</sup><https://xainlp2020.github.io/xainlp/>  
(we plan to maintain this website as a contribution to the community.)

paper covered in this survey can be found at the aforementioned website; to enable readers of this survey to discover interesting explainability techniques and ideas, even if they have not been fully developed in the respective publications.

## 3 Categorization of Explanations

Explanations are often categorized along two main aspects (Guidotti et al., 2018; Adadi and Berrada, 2018). The first distinguishes whether the explanation is for an individual prediction (*local*) or the model’s prediction process as a whole (*global*). The second differentiates between the explanation emerging directly from the prediction process (*self-explaining*) versus requiring post-processing (*post-hoc*). We next describe both of these aspects in detail, and provide a summary of the four categories they induce in Table 1.

### 3.1 Local vs Global

A *local* explanation provides information or justification for the model’s prediction on a specific input; 46 of the 50 papers fall into this category.

A *global* explanation provides similar justification by revealing how the model’s predictive process works, independently of any particular input. This category holds the remaining 4 papers covered by this survey. This low number is not surprising given the focus of this survey being on explanations that justify predictions, as opposed to explanations that help understand a model’s behavior in general (which lie outside the scope of this survey).

### 3.2 Self-Explaining vs Post-Hoc

Regardless of whether the explanation is local or global, explanations differ on whether they arise as part of the prediction process, or whether their generation requires post-processing following the model making a prediction. A *self-explaining* approach, which may also be referred to as directly interpretable (Arya et al., 2019), generates the explanation at the same time as the prediction, using information emitted by the model as a result of the process of making that prediction. Decision trees and rule-based models are examples of global self-explaining models, while feature saliency approaches such as attention are examples of local self-explaining models.

In contrast, a *post-hoc* approach requires that an additional operation is performed after the predictions are made. LIME (Ribeiro et al., 2016) is

an example of producing a local explanation using a surrogate model applied following the predictor’s operation. A paper might also be considered to span both categories – for example, ([Sydorova et al., 2019](#)) actually presents both self-explaining and post-hoc explanation techniques.

<b>Local Post-Hoc</b>	Explain a single prediction by performing additional operations ( <i>after</i> the model has emitted a prediction)
<b>Local Self-Explaining</b>	Explain a single prediction using the model itself (calculated from information made available from the model <i>as part of</i> making the prediction)
<b>Global Post-Hoc</b>	Perform additional operations to explain the entire model’s predictive reasoning
<b>Global Self-Explaining</b>	Use the predictive model itself to explain the entire model’s predictive reasoning ( <i>a.k.a.</i> directly interpretable model)

Table 1: Overview of the high-level categories of explanations (Section 3).

## 4 Aspects of Explanations

While the previous categorization serves as a convenient high-level classification of explanations, it does not cover other important characteristics. We now introduce two additional aspects of explanations: (1) techniques for deriving the explanation and (2) presentation to the end user. We discuss the most commonly used explainability techniques, along with basic operations that enable explainability, as well as the visualization techniques commonly used to present the output of associated explainability techniques. We identify the most common combinations of explainability techniques, operations, and visualization techniques for each of the four high-level categories of explanations presented above, and summarize them, together with representative papers, in Table 2.

Although explainability techniques and visualizations are often intermixed, there are fundamental differences between them that motivated us to treat them separately. Concretely, explanation derivation - typically done by AI scientists and engineers - focuses on mathematically motivated justifications of models’ output, leveraging various explainability techniques to produce “raw explanations” (such as attention scores). On the other hand, explanation presentation - ideally done by UX engineers - focuses on how these “raw explanations” are best presented to the end users using suitable visualization techniques (such as saliency heatmaps).

### 4.1 Explainability Techniques

In the papers surveyed, we identified five major explainability techniques that differ in the mechanisms they adopt to generate the raw mathematical justifications that lead to the final explanation presented to the end users.

*Feature importance.* The main idea is to derive explanation by investigating the importance scores of different features used to output the final prediction. Such approaches can be built on different types of features, such as manual features obtained from feature engineering (e.g., [Voskarides et al., 2015](#)), lexical features including word/tokens and n-gram (e.g., [Godin et al., 2018; Mullenbach et al., 2018](#)), or latent features learned by NNs (e.g., [Xie et al., 2017](#)). Attention mechanism ([Bahdanau et al., 2015](#)) and first-derivative saliency ([Li et al., 2015](#)) are two widely used operations to enable feature importance-based explanations. Text-based features are inherently more interpretable by humans than general features, which may explain the widespread use of attention-based approaches in the NLP domain.

*Surrogate model.* Model predictions are explained by learning a second, usually more explainable model, as a proxy. One well-known example is LIME ([Ribeiro et al., 2016](#)), which learns surrogate models using an operation called input perturbation. Surrogate model-based approaches are model-agnostic and can be used to achieve either local (e.g., [Alvarez-Melis and Jaakkola, 2017](#)) or global (e.g., [Liu et al., 2018](#)) explanations. However, the learned surrogate models and the original models may have completely different mechanisms to make predictions, leading to concerns about the fidelity of surrogate model-based approaches.

*Example-driven.* Such approaches explain the prediction of an input instance by identifying and presenting other instances, usually from available labeled data, that are semantically similar to the input instance. They are similar in spirit to nearest neighbor-based approaches ([Dudani, 1976](#)), and have been applied to different NLP tasks such as text classification ([Croce et al., 2019](#)) and question answering ([Abujabal et al., 2017](#)).

*Provenance-based.* Explanations are provided by illustrating some or all of the prediction derivation process, which is an intuitive and effective explainability technique when the final prediction is the result of a series of reasoning steps. We observe several question answering papers adopt such ap-

Category (#)	Explainability Technique	Operations to Enable Explainability	Visualization Technique	#	Representative Paper(s)
<b>Local Post-Hoc (11)</b>	feature importance	first derivative saliency, example driven	saliency	5	(Wallace et al., 2018; Ross et al., 2017)
	surrogate model	first derivative saliency, layer-wise relevance propagation, input perturbation	saliency	4	(Alvarez-Melis and Jaakkola, 2017; Poerner et al., 2018; Ribeiro et al., 2016)
	example driven	layer-wise relevance propagation, explainability-aware architecture	raw examples	2	(Croce et al., 2018; Jiang et al., 2019)
<b>Local Self-Exp (35)</b>	feature importance	attention, first derivative saliency, LSTM gating signals, explainability-aware architecture	saliency	22	(Mullenbach et al., 2018; Ghaeini et al., 2018; Xie et al., 2017; Aubakirova and Bansal, 2016)
	induction	explainability-aware architecture, rule induction	raw declarative representation	6	(Ling et al., 2017; Dong et al., 2019; Pezeshkpour et al., 2019a)
	provenance	template-based	natural language, other	3	(Abujabal et al., 2017)
	surrogate model	attention, input perturbation, explainability-aware architecture	natural language	3	(Rajani et al., 2019a; Sydorova et al., 2019)
<b>Global Post-Hoc (3)</b>	example driven	layer-wise relevance propagation	raw examples	1	(Croce et al., 2019)
	feature importance	class activation mapping, attention, gradient reversal	saliency	2	(Pryzant et al., 2018a,b)
	surrogate model	taxonomy induction	raw declarative representation	1	(Liu et al., 2018)
<b>Global Self-Exp (1)</b>	induction	reinforcement learning	raw declarative representation	1	(Pröllochs et al., 2019)

Table 2: Overview of common combinations of explanation aspects: columns 2, 3, and 4 capture explainability techniques, operations, and visualization techniques, respectively (see Sections 4.1, 4.2, and 4.3 for details). These are grouped by the high-level categories detailed in Section 3, as shown in the first column. The last two columns show the number of papers in this survey that fall within each subgroup, and a list of representative references.

proaches (Abujabal et al., 2017; Zhou et al., 2018; Amini et al., 2019).

*Declarative induction.* Human-readable representations, such as rules (Pröllochs et al., 2019), trees (Voskarides et al., 2015), and programs (Ling et al., 2017) are induced as explanations.

As shown in Table 2, feature importance-based and surrogate model-based approaches have been in frequent use (accounting for 29 and 8, respectively, of the 50 papers reviewed). This should not come as a surprise, as features serve as building blocks for machine learning models (explaining the proliferation of feature importance-based approaches) and most recent NLP papers employ NN-based models, which are generally black-box models (explaining the popularity of surrogate model-based approaches). Finally note that a complex NLP approach consisting of different components

may employ more than one of these explainability techniques. A representative example is the QA system QUINT (Abujabal et al., 2017), which displays the query template that best matches the user input query (example-driven) as well as the instantiated knowledge-base entities (provenance).

## 4.2 Operations to Enable Explainability

We now present the most common set of operations encountered in our literature review that are used to enable explainability, in conjunction with relevant work employing each one.

*First-derivative saliency.* Gradient-based explanations estimate the contribution of input  $i$  towards output  $o$  by computing the partial derivative of  $o$  with respect to  $i$ . This is closely related to older concepts such as sensitivity (Saltelli et al., 2008). First-derivative saliency is particularly con-

venient for NN-based models because these can be computed for any layer using a single call to auto-differentiation, which most deep learning engines provide out-of-the-box. Recent work has also proposed improvements to first-derivative saliency (Sundararajan et al., 2017). As suggested by its name and definition, first-derivative saliency can be used to enable feature importance explainability, especially on word/token-level features (Aubakirova and Bansal, 2016; Karlekar et al., 2018).

*Layer-wise relevance propagation.* This is another way to attribute relevance to features computed in any intermediate layer of an NN. Definitions are available for most common NN layers including fully connected layers, convolution layers and recurrent layers. Layer-wise relevance propagation has been used to, for example, enable feature importance explainability (Poerner et al., 2018) and example-driven explainability (Croce et al., 2018).

*Input perturbations.* Pioneered by LIME (Ribeiro et al., 2016), input perturbations can explain the output for input  $x$  by generating random perturbations of  $x$  and training an explainable model (usually a linear model). They are mainly used to enable surrogate models (e.g., Ribeiro et al., 2016; Alvarez-Melis and Jaakkola, 2017).

*Attention* (Bahdanau et al., 2015; Vaswani et al., 2017). Less an operation and more of a strategy to enable the NN to explain predictions, attention layers can be added to most NN architectures and, because they appeal to human intuition, can help indicate where the NN model is “focusing”. While previous work has widely used attention layers (Luo et al., 2018; Xie et al., 2017; Mullenbach et al., 2018) to enable feature importance explainability, the jury is still out as to how much explainability attention provides (Jain and Wallace, 2019; Serrano and Smith, 2019; Wiegreffe and Pinter, 2019).

*LSTM gating signals.* Given the sequential nature of language, recurrent layers, in particular LSTMs (Hochreiter and Schmidhuber, 1997), are commonplace. While it is common to mine the outputs of LSTM cells to explain outputs, there may also be information present in the outputs of the gates produced within the cells. It is possible to utilize (and even combine) other operations presented here to interpret gating signals to aid feature importance explainability (Ghaeini et al., 2018).

*Explainability-aware architecture design.* One way to exploit the flexibility of deep learning is to devise an NN architecture that mimics the process

humans employ to arrive at a solution. This makes the learned model (partially) interpretable since the architecture contains human-recognizable components. Implementing such a model architecture can be used to enable the induction of human-readable programs for solving math problems (Amini et al., 2019; Ling et al., 2017) or sentence simplification problems (Dong et al., 2019). This design may also be applied to surrogate models that generate explanations for predictions (Rajani et al., 2019a; Liu et al., 2019).

Previous works have also attempted to compare these operations in terms of efficacy with respect to specific NLP tasks (Poerner et al., 2018). Operations outside of this list exist and are popular for particular categories of explanations. Table 2 mentions some of these. For instance, Pröllochs et al. (2019) use reinforcement learning to learn simple negation rules, Liu et al. (2018) learns a taxonomy post-hoc to better interpret network embeddings, and Pryzant et al. (2018b) uses gradient reversal (Ganin et al., 2016) to deconfound lexicons.

### 4.3 Visualization Techniques

An explanation may be presented in different ways to the end user, and making the appropriate choice is crucial for the overall success of an XAI approach. For example, the widely used attention mechanism, which learns the importance scores of a set of features, can be visualized as raw attention scores or as a saliency heatmap (see Figure 1a). Although the former is acceptable, the latter is more user-friendly and has become the standard way to visualize attention-based approaches. We now present the major visualization techniques identified in our literature review.

*Saliency.* This has been primarily used to visualize the importance scores of different types of elements in XAI learning systems, such as showing input-output word alignment (Bahdanau et al., 2015) (Figure 1a), highlighting words in input text (Mullenbach et al., 2018) (Figure 1b) or displaying extracted relations (Xie et al., 2017). We observe a strong correspondence between feature importance-based explainability and saliency-based visualizations; namely, all papers using feature importance to generate explanations also chose saliency-based visualization techniques. Saliency-based visualizations are popular because they present visually perceptive explanations and can be easily understood by different types of end users. They are there-

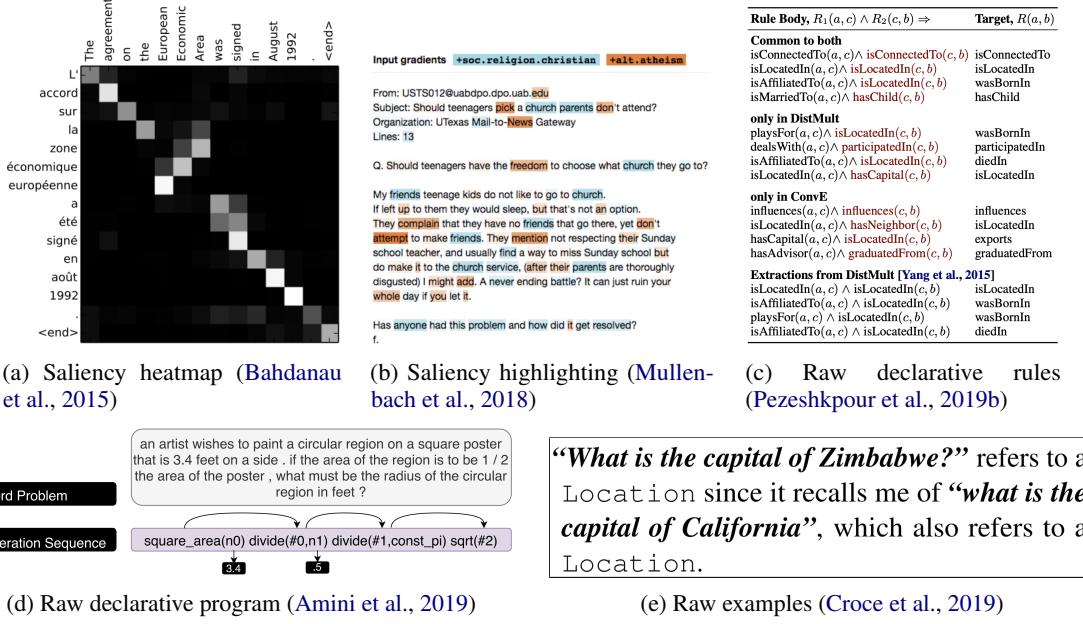


Figure 1: Examples of different visualization techniques

fore frequently seen across different AI domains (e.g., computer vision (Simonyan et al., 2013) and speech (Aldeneh and Provost, 2017)). As shown in Table 2, saliency is the most dominant visualization technique among the papers covered by this survey.

*Raw declarative representations.* As suggested by its name, this visualization technique directly presents the learned declarative representations, such as logic rules, trees, and programs (Figure 1c and 1d). Such techniques assume that end users can understand specific representations, such as first-order logic rules (Pezeshkpour et al., 2019a) and reasoning trees (Liang et al., 2016), and therefore may implicitly target more advanced users.

*Natural language explanation.* The explanation is verbalized in human-comprehensible natural language (Figure 2). The natural language can be generated using sophisticated deep learning models, e.g., by training a language model with human natural language explanations and coupling with a deep generative model (Rajani et al., 2019a). It can also be generated by using simple template-based approaches (Abujabal et al., 2017). In fact, many declarative induction-based techniques can use template-based natural language generation (Reiter and Dale, 1997) to turn rules and programs into human-comprehensible language, and this minor extension can potentially make the explanation more accessible to lay users.

Table 2 references some additional visualization techniques, such as using *raw examples* to

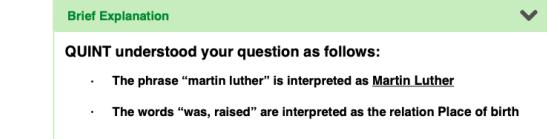


Figure 2: Template-based natural language explanation for a QA system (Abujabal et al., 2017).

present example-driven approaches (Jiang et al., 2019; Croce et al., 2019) (e.g., Figure 1e), and dependency parse trees to represent input questions (Abujabal et al., 2017).

## 5 Explanation Quality

Following the goals of XAI, a model’s quality should be evaluated not only by its accuracy and performance, but also by how well it provides explanations for its predictions. In this section we discuss the state of the field in terms of defining and measuring explanation quality.

### 5.1 Evaluation

Given the young age of the field, unsurprisingly there is little agreement on how explanations should be evaluated. The majority of the works reviewed (32 out of 50) either lack a standardized evaluation or include only an informal evaluation, while a smaller number of papers looked at more formal evaluation approaches, including leveraging ground truth data and human evaluation. We next present the major categories of evaluation tech-

niques we encountered (summarized in Table 3).

None or Informal Examination only	Comparison to Ground Truth	Human Evaluation
32	12	9

Table 3: Common evaluation techniques and number of papers adopting them, out of the 50 papers surveyed (note that some papers adopt more than one technique)

*Informal examination* of explanations. This typically takes the form of high-level discussions of how examples of generated explanations align with human intuition. This includes cases where the output of a single explainability approach is examined in isolation (Xie et al., 2017) as well as when explanations are compared to those of other reference approaches (Ross et al., 2017) (such as LIME, which is a frequently used baseline).

*Comparison to ground truth.* Several works compare generated explanations to ground truth data in order to quantify the performance of explainability techniques. Employed metrics vary based on task and explainability technique, but commonly encountered metrics include P/R/F1 (Carton et al., 2018), perplexity, and BLEU (Ling et al., 2017; Rajani et al., 2019b). While having a quantitative way to measure explainability is a promising direction, care should be taken during ground truth acquisition to ensure its quality and account for cases where there may be alternative valid explanations. Approaches employed to address this issue involve having multiple annotators and reporting inter-annotator agreement or mean human performance, as well as evaluating the explanations at different granularities (e.g., token-wise vs phrase-wise) to account for disagreements on the precise value of the ground truth (Carton et al., 2018).

*Human evaluation.* A more direct way to assess the explanation quality is to ask humans to evaluate the effectiveness of the generated explanations. This has the advantage of avoiding the assumption that there is only one good explanation that could serve as ground truth, as well as sidestepping the need to measure similarity of explanations. Here as well, it is important to have multiple annotators, report inter-annotator agreement, and correctly deal with subjectivity and variance in the responses. The approaches found in this survey vary in several dimensions, including the number of humans involved (ranging from 1 (Mullenbach et al., 2018) to 25 (Sydorova et al., 2019) humans), as well as the

high-level task that they were asked to perform (including rating the explanations of a single approach (Dong et al., 2019) and comparing explanations of multiple techniques (Sydorova et al., 2019)).

*Other operation-specific techniques.* Given the prevalence of attention layers (Bahdanau et al., 2015; Vaswani et al., 2017) in NLP, recent work (Jain and Wallace, 2019; Serrano and Smith, 2019; Wiegreffe and Pinter, 2019) has developed specific techniques to evaluate such explanations based on counterfactuals or erasure-based tests (Feng et al., 2018). Serrano and Smith repeatedly set to zero the maximal entry produced by the attention layer. If attention weights indeed “explain” the output prediction, then turning off the dominant weights should result in an altered prediction. Similar experiments have been devised by others (Jain and Wallace, 2019). In particular, Wiegreffe and Pinter caution against assuming that there exists only one true explanation to suggest accounting for the natural variance of attention layers. On a broader note, causality has thoroughly explored such counterfactual-based notions of explanation (Halpern, 2016).

While the above overview summarizes *how* explainability approaches are commonly evaluated, another important aspect is *what* is being evaluated. Explanations are multi-faceted objects that can be evaluated on multiple aspects, such as *fidelity* (how much they reflect the actual workings of the underlying model), *comprehensibility* (how easy they are to understand by humans), and others. Therefore, understanding the target of the evaluation is important for interpreting the evaluation results. We refer interested readers to (Carvalho et al., 2019) for a comprehensive presentation of aspects of evaluating approaches.

Many works do not explicitly state what is being evaluated. As a notable exception, (Lertvittayakumjorn and Toni, 2019) outlines three goals of explanations (reveal model behavior, justify model predictions, and assist humans in investigating uncertain predictions) and proposes human evaluation experiments targeting each of them.

## 5.2 Predictive Process Coverage

An important and often overlooked aspect of explanation quality is the part of the prediction process (starting with the input and ending with the model output) covered by an explanation. We have observed that many explainability approaches explain only part of this process, leaving it up to the end

user to fill in the gaps.

As an example, consider the MathQA task of solving math word problems. As readers may be familiar from past education experience, in math exams, one is often asked to provide a step-by-step explanation of how the answer was derived. Usually, full credit is not given if any of the critical steps used in the derivation are missing. Recent works have studied the explainability of MathQA models, which seek to reproduce this process (Amini et al., 2019; Ling et al., 2017), and have employed different approaches in the type of explanations produced. While (Amini et al., 2019) explains the predicted answer by showing the sequence of mathematical operations leading to it, this provides only partial coverage, as it does not explain how these operations were derived from the input text. On the other hand, the explanations produced by (Ling et al., 2017) augment the mathematical formulas with text describing *the thought process* behind the derived solution, thus covering a bigger part of the prediction process.

The level of coverage may be an artifact of explainability techniques used: provenance-based approaches tend to provide more coverage, while example-driven approaches, may provide little to no coverage. Moreover, while our math teacher would argue that providing higher coverage is always beneficial to the student, in reality this may depend on the end use of the explanation. For instance, the coverage of explanations of (Amini et al., 2019) may be potentially sufficient for advanced technical users. Thus, higher coverage, while in general a positive aspect, should always be considered in combination with the target use and audience of the produced explanations.

## 6 Insights and Future Directions

This survey showcases recent advances of XAI research in NLP, as evidenced by publications in major NLP conferences in the last 7 years. We have discussed the main categorization of explanations (Local vs Global, Self-Explaining vs Post-Hoc) as well as the various ways explanations can be arrived at and visualized, together with the common techniques used. We have also detailed operations and explainability techniques currently available for generating explanations of model predictions, in the hopes of serving as a resource for developers interested in building explainable NLP models.

We hope this survey encourages the research

community to work in bridging the current gaps in the field of XAI in NLP. The first research direction is a need for clearer terminology and understanding of what constitutes explainability and how it connects to the target audience. For example, is a model that displays an induced program that, when executed, yields a prediction, and yet conceals the process of inducing the program, explainable in general? Or is it explainable for some target users but not for others? The second is an expansion of the evaluation processes and metrics, especially for human evaluation. The field of XAI is aimed at adding explainability as a desired feature of models, in addition to the model’s predictive quality, and other features such as runtime performance, complexity or memory usage. In general, trade-offs exist between desired characteristics of models, such as more complex models achieving better predictive power at the expense of slower runtime. In XAI, some works have claimed that explainability may come at the price of losing predictive quality (Bertsimas et al., 2019), while others have claimed the opposite (Garneau et al., 2018; Liang et al., 2016). Studying such possible trade-offs is an important research area for XAI, but one that cannot advance until standardized metrics are developed for evaluating the quality of explanations. The third research direction is a call to more critically address the issue of fidelity (or causality), and to ask hard questions about whether a claimed explanation is faithfully explaining the model’s prediction.

Finally, it is interesting to note that we found only four papers that fall into the global explanations category. This might seem surprising given that white box models, which have been fundamental in NLP, are explainable in the global sense. We believe this stems from the fact that because white box models are clearly explainable, the focus of the explicit XAI field is in explaining black box models, which comprise mostly local explanations. White box models, like rule based models and decision trees, while still in use, are less frequently framed as explainable or interpretable, and are hence not the main thrust of where the field is going. We think that this may be an oversight of the field since white box models can be a great test bed for studying techniques for evaluating explanations.

## Acknowledgments

We thank the anonymous reviewers for their valuable feedback. We also thank Shipi Dhanorkar,

Yunyao Li, Lucian Popa, Christine T Wolf, and Anbang Xu for their efforts at the early stage of this work.

## References

- Abdalghani Abujabal, Rishiraj Saha Roy, Mohamed Yahya, and Gerhard Weikum. 2017. Quint: Interpretable question answering over knowledge bases. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 61–66.
- A. Adadi and M. Berrada. 2018. Peeking inside the black-box: A survey on explainable artificial intelligence (xai). *IEEE Access*, 6:52138–52160.
- Zakaria Aldeneh and Emily Mower Provost. 2017. Using regional saliency for speech emotion recognition. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2741–2745. IEEE.
- David Alvarez-Melis and Tommi Jaakkola. 2017. A causal framework for explaining the predictions of black-box sequence-to-sequence models. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 412–421, Copenhagen, Denmark. Association for Computational Linguistics.
- Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. MathQA: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2357–2367, Minneapolis, Minnesota. Association for Computational Linguistics.
- Vijay Arya, Rachel K. E. Bellamy, Pin-Yu Chen, Amit Dhurandhar, Michael Hind, Samuel C. Hoffman, Stephanie Houde, Q. Vera Liao, Ronny Luss, Aleksandra Mojsilovic, Sami Mourad, Pablo Pedemonte, Ramya Raghavendra, John T. Richards, Prasanna Sattigeri, Karthikeyan Shanmugam, Moninder Singh, Kush R. Varshney, Dennis Wei, and Yi Zhang. 2019. One explanation does not fit all: A toolkit and taxonomy of ai explainability techniques. *ArXiv*, abs/1909.03012.
- M. Aubakirova and M. Bansal. 2016. Interpreting neural networks to improve politeness comprehension. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (Austin, Texas, 2016)*, page 2035–2041.
- AmirAli Bagher Zadeh, Paul Pu Liang, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. 2018. Multimodal language analysis in the wild: CMU-MOSEI dataset and interpretable dynamic fusion graph. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2236–2246, Melbourne, Australia. Association for Computational Linguistics.
- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.
- Francesco Barbieri, Luis Espinosa-Anke, Jose Camacho-Collados, Steven Schockaert, and Horacio Saggion. 2018. Interpretable emoji prediction via label-wise attention LSTMs. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4766–4771, Brussels, Belgium. Association for Computational Linguistics.
- Dimitris Bertsimas, Arthur Delarue, Patrick Jaillet, and Sébastien Martin. 2019. The price of interpretability. *ArXiv*, abs/1907.03419.
- Nikita Bhutani, Kun Qian, Yunyao Li, H. V. Jagadish, Mauricio Hernandez, and Mitesh Vasa. 2018. Exploiting structure in representation of named entities using active learning. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 687–699, Santa Fe, New Mexico, USA. Association for Computational Linguistics.
- Samuel Carton, Qiaozhu Mei, and Paul Resnick. 2018. Extractive adversarial networks: High-recall explanations for identifying personal attacks in social media posts. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3497–3507, Brussels, Belgium. Association for Computational Linguistics.
- Diogo V. Carvalho, Eduardo M. Pereira, and Jaime S. Cardoso. 2019. Machine Learning Interpretability: A Survey on Methods and Metrics. *Electronics*, 8(8):832. Number: 8 Publisher: Multidisciplinary Digital Publishing Institute.
- Danilo Croce, Daniele Rossini, and Roberto Basili. 2018. Explaining non-linear classifier decisions within kernel-based deep architectures. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 16–24, Brussels, Belgium. Association for Computational Linguistics.
- Danilo Croce, Daniele Rossini, and Roberto Basili. 2019. Auditing deep learning processes through kernel-based explanatory models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4037–4046, Hong Kong, China. Association for Computational Linguistics.
- Yue Dong, Zichao Li, Mehdi Rezagholizadeh, and Jackie Chi Kit Cheung. 2019. EditNTS: An neural programmer-interpreter model for sentence simplification through explicit editing. In *Proceedings of*

- the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3393–3402, Florence, Italy. Association for Computational Linguistics.
- Sahibsingh A Dudani. 1976. The distance-weighted k-nearest-neighbor rule. *IEEE Transactions on Systems, Man, and Cybernetics*, (4):325–327.
- Shi Feng, Eric Wallace, Alvin Grissom II, Mohit Iyyer, Pedro Rodriguez, and Jordan Boyd-Graber. 2018. Pathologies of neural models make interpretations difficult. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3719–3728, Brussels, Belgium. Association for Computational Linguistics.
- Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, , and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *JMLR*.
- Nicolas Garneau, Jean-Samuel Leboeuf, and Luc Lamontagne. 2018. Predicting and interpreting embeddings for out of vocabulary words in downstream tasks. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 331–333, Brussels, Belgium. Association for Computational Linguistics.
- Reza Ghaeini, Xiaoli Fern, and Prasad Tadepalli. 2018. Interpreting recurrent and attention-based neural models: a case study on natural language inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4952–4957, Brussels, Belgium. Association for Computational Linguistics.
- Frédéric Godin, Kris Demuynck, Joni Dambre, Wesley De Neve, and Thomas Demeester. 2018. Explaining character-aware neural networks for word-level prediction: Do they discover linguistic rules? In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3275–3284, Brussels, Belgium. Association for Computational Linguistics.
- Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. 2018. A survey of methods for explaining black box models. *ACM Comput. Surv.*, 51(5).
- Pankaj Gupta and Hinrich Schütze. 2018. LISA: Explaining recurrent neural network judgments via layer-wise semantic accumulation and example to pattern transformation. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 154–164, Brussels, Belgium. Association for Computational Linguistics.
- Joseph Y. Halpern. 2016. *Actual Causality*. MIT Press.
- David Harbecke, Robert Schwarzenberg, and Christoph Alt. 2018. Learning explanations from language data. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 316–318, Brussels, Belgium. Association for Computational Linguistics.
- Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*.
- Shiou Tian Hsu, Changsung Moon, Paul Jones, and Nagiza Samatova. 2018. An interpretable generative adversarial approach to classification of latent entity relations in unstructured sentences. In *AAAI Conference on Artificial Intelligence*.
- Sarthak Jain and Byron C. Wallace. 2019. Attention is not Explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3543–3556, Minneapolis, Minnesota. Association for Computational Linguistics.
- Yichen Jiang and Mohit Bansal. 2019. Self-assembling modular networks for interpretable multi-hop reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4474–4484, Hong Kong, China. Association for Computational Linguistics.
- Yichen Jiang, Nitish Joshi, Yen-Chun Chen, and Mohit Bansal. 2019. Explore, propose, and assemble: An interpretable model for multi-hop reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2714–2725, Florence, Italy. Association for Computational Linguistics.
- Dongyeop Kang, Varun Gangal, Ang Lu, Zheng Chen, and Eduard Hovy. 2017. Detecting and explaining causes from text for a time series event. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2758–2767, Copenhagen, Denmark. Association for Computational Linguistics.
- Sweta Karlekar, Tong Niu, and Mohit Bansal. 2018. Detecting linguistic characteristics of alzheimer’s dementia by interpreting neural models. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers) (New Orleans, Louisiana, Jun. 2018)*, page 701–707.
- Shun Kiyono, Sho Takase, Jun Suzuki, Naoaki Okazaki, Kentaro Inui, and Masaaki Nagata. 2018. Unsupervised token-wise alignment to improve interpretation of encoder-decoder models. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 74–81, Brussels, Belgium. Association for Computational Linguistics.

- Freddy Lecue, Krishna Gade, Sahin Cem Geyik, Krishnamoorthy Kenthapadi, Varun Mithal, Ankur Taly, Ricardo Guidotti, and Pasquale Minervini. 2020. **Explainable ai: Foundations, industrial applications, practical challenges, and lessons learned**. In *AAAI Conference on Artificial Intelligence*. Association for Computational Linguistics.
- Piyawat Lertvittayakumjorn and Francesca Toni. 2019. **Human-grounded evaluations of explanation methods for text classification**. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5195–5205, Hong Kong, China. Association for Computational Linguistics.
- Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. 2015. Visualizing and understanding neural models in nlp. *arXiv preprint arXiv:1506.01066*.
- Qiuchi Li, Benyou Wang, and Massimo Melucci. 2019. **CNM: An interpretable complex-valued network for matching**. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4139–4148, Minneapolis, Minnesota. Association for Computational Linguistics.
- Chao-Chun Liang, Shih-Hong Tsai, Ting-Yun Chang, Yi-Chung Lin, and Keh-Yih Su. 2016. **A meaning-based English math word problem solver with understanding, reasoning and explanation**. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*, pages 151–155, Osaka, Japan. The COLING 2016 Organizing Committee.
- Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. **Program induction by rationale generation: Learning to solve and explain algebraic word problems**. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 158–167, Vancouver, Canada. Association for Computational Linguistics.
- Hui Liu, Qingyu Yin, and William Yang Wang. 2019. **Towards explainable NLP: A generative explanation framework for text classification**. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5570–5581, Florence, Italy. Association for Computational Linguistics.
- Ninghao Liu, Xiao Huang, Jundong Li, and Xia Hu. 2018. **On interpretation of network embedding via taxonomy induction**. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD ’18, page 1812–1820, New York, NY, USA. Association for Computing Machinery.
- Junyu Lu, Chenbin Zhang, Zeying Xie, Guang Ling, Tom Chao Zhou, and Zenglin Xu. 2019. **Constructing interpretive spatio-temporal features for multi-turn responses selection**. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 44–50, Florence, Italy. Association for Computational Linguistics.
- Ling Luo, Xiang Ao, Feiyang Pan, Jin Wang, Tong Zhao, Ningzi Yu, and Qing He. 2018. Beyond polarity: Interpretable financial sentiment analysis with hierarchical query-driven attention.
- Seungwhan Moon, Pararth Shah, Anuj Kumar, and Rajen Subba. 2019. **OpenDialKG: Explainable conversational reasoning with attention-based walks over knowledge graphs**. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 845–854, Florence, Italy. Association for Computational Linguistics.
- James Mullenbach, Sarah Wiegreffe, Jon Duke, Jimeng Sun, and Jacob Eisenstein. 2018. **Explainable prediction of medical codes from clinical text**. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1101–1111, New Orleans, Louisiana. Association for Computational Linguistics.
- An Nguyen, Aditya Kharosekar, Matthew Lease, and Byron Wallace. 2018. **An interpretable joint graphical model for fact-checking from crowds**. In *AAAI Conference on Artificial Intelligence*.
- Alexander Panchenko, Fide Marten, Eugen Ruppert, Stefano Faralli, Dmitry Ustalov, Simone Paolo Ponzetto, and Chris Biemann. 2017. **Unsupervised, knowledge-free, and interpretable word sense disambiguation**. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 91–96, Copenhagen, Denmark. Association for Computational Linguistics.
- Nikolaos Pappas and Andrei Popescu-Belis. 2014. **Explaining the stars: Weighted multiple-instance learning for aspect-based sentiment analysis**. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 455–466, Doha, Qatar. Association for Computational Linguistics.
- Pouya Pezeshkpour, Yifan Tian, and Sameer Singh. 2019a. **Investigating robustness and interpretability of link prediction via adversarial modifications**. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3336–3347, Minneapolis, Minnesota. Association for Computational Linguistics.
- Pouya Pezeshkpour, Yifan Tian, and Sameer Singh. 2019b. **Investigating robustness and interpretability of link prediction via adversarial modifications**. In

- Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3336–3347.
- Nina Poerner, Hinrich Schütze, and Benjamin Roth. 2018. Evaluating neural network explanation methods using hybrid documents and morphosyntactic agreement. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 340–350, Melbourne, Australia. Association for Computational Linguistics.
- Nicolas Pröllochs, Stefan Feuerriegel, and Dirk Neumann. 2019. Learning interpretable negation rules via weak supervision at document level: A reinforcement learning approach. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 407–413, Minneapolis, Minnesota. Association for Computational Linguistics.
- Reid Pryzant, Sugato Basu, and Kazoo Sone. 2018a. Interpretable neural architectures for attributing an ad’s performance to its writing style. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 125–135, Brussels, Belgium. Association for Computational Linguistics.
- Reid Pryzant, Kelly Shen, Dan Jurafsky, and Stefan Wagner. 2018b. Deconfounded lexicon induction for interpretable social science. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1615–1625, New Orleans, Louisiana. Association for Computational Linguistics.
- Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019a. Explain yourself! leveraging language models for commonsense reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4932–4942, Florence, Italy. Association for Computational Linguistics.
- Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019b. Explain yourself! leveraging language models for commonsense reasoning. *arXiv preprint arXiv:1906.02361*.
- Ehud Reiter and Robert Dale. 1997. Building applied natural language generation systems. *Natural Language Engineering*, 3(1):57–87.
- Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. Why should i trust you??: Explaining the predictions of any classifier. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (New York, NY, USA, 2016)*, page 1135–1144.
- Andrew Slavin Ross, Michael C. Hughes, and Finale Doshi-Velez. 2017. Right for the right reasons: Training differentiable models by constraining their explanations. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 2662–2670.
- A. Saltelli, M. Ratto, T. Andres, F. Campolongo, J. Cariboni, D. Gatelli, M. Saisana, and S. Tarantola. 2008. *Global Sensitivity Analysis: The Primer*. John Wiley & Sons.
- Robert Schwarzenberg, David Harbecke, Vivien Macketanz, Eleftherios Avramidis, and Sebastian Möller. 2019. Train, sort, explain: Learning to diagnose translation models. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 29–34, Minneapolis, Minnesota. Association for Computational Linguistics.
- Prithviraj Sen, Yunyao Li, Eser Kandogan, Yiwei Yang, and Walter Lasecki. 2019. HEIDL: Learning linguistic expressions with deep learning and human-in-the-loop. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 135–140, Florence, Italy. Association for Computational Linguistics.
- Sofia Serrano and Noah A. Smith. 2019. Is attention interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2931–2951, Florence, Italy. Association for Computational Linguistics.
- Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*.
- Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, Sydney, Australia.
- Alona Sydorova, Nina Poerner, and Benjamin Roth. 2019. Interpretable question answering on knowledge bases and text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4943–4951, Florence, Italy. Association for Computational Linguistics.
- James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2019. Generating token-level explanations for natural language inference. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 963–969, Minneapolis, Minnesota. Association for Computational Linguistics.
- Martin Tutek and Jan Šnajder. 2018. Iterative recursive attention model for interpretable sequence classification. In *Proceedings of the 2018 EMNLP Work-*

shop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, pages 249–257, Brussels, Belgium. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*.

Nikos Voskarides, Edgar Meij, Manos Tsagkias, Maarten de Rijke, and Wouter Weerkamp. 2015. Learning to explain entity relationships in knowledge graphs. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 564–574, Beijing, China. Association for Computational Linguistics.

Eric Wallace, Shi Feng, and Jordan Boyd-Graber. 2018. Interpreting neural networks with nearest neighbors. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 136–144, Brussels, Belgium. Association for Computational Linguistics.

Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 11–20, Hong Kong, China. Association for Computational Linguistics.

Qizhe Xie, Xuezhe Ma, Zihang Dai, and Eduard Hovy. 2017. An interpretable knowledge transfer model for knowledge base completion. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 950–962, Vancouver, Canada. Association for Computational Linguistics.

Yang Yang, Deyu Zhou, Yulan He, and Meng Zhang. 2019. Interpretable relevant emotion ranking with event-driven attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 177–187, Hong Kong, China. Association for Computational Linguistics.

Mantong Zhou, Minlie Huang, and Xiaoyan Zhu. 2018. An interpretable reasoning network for multi-relation question answering. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2010–2022.

## A Appendix A - Methodology

This survey aims to demonstrate the recent advances of XAI research in NLP, rather than to provide an exhaustive list of XAI papers in the NLP community. To this end, we identified relevant papers published in major NLP conferences (ACL,

NAACL, EMNLP, and COLING) between 2013 and 2019. We filtered for titles containing (lemmatized) terms related to XAI, such as “explainability”, “interpretability”, “transparent”, etc. While this may ignore some related papers, we argue that representative papers are more likely to include such terms in their titles. In particular, we assume that if authors consider explainability to be a major component of their work, they are more likely to use related keywords in the title of their work. Our search criteria yielded a set of 107 papers.

Top 3 NLP Topics		
1	2	3
Question Answering (9)	Computational Social Science & Social Media (6)	Syntax: Tagging, Chunking & Parsing (6)
Top 3 Conferences		
1	2	3
EMNLP (21)	ACL (12)	NAACL (9)

Table 4: Top NLP topics and conferences (2013-2019) of papers included in this survey

During the paper review process we first verified whether each paper truly fell within the scope of the survey; namely, papers with a focus on explainability as a vehicle for understanding how a model arrives at its result. This process excluded 57 papers, leaving us with a total of 50 papers. Table 4 lists the top three broad NLP topics (taken verbatim from the ACL call for papers) covered by these 50 papers, and the top three conferences of the set.

To ensure a consistent classification, each paper was individually reviewed by at least two reviewers, consulting additional reviewers in the case of disagreement.

# Beyond Fine-tuning: Few-Sample Sentence Embedding Transfer

Siddhant Garg<sup>\*†</sup>

Amazon Alexa AI Search

Manhattan Beach, CA, USA

sidgarg@amazon.com

Rohit Kumar Sharma<sup>\*†</sup>

Microsoft

Seattle, WA, USA

rsharma@cs.wisc.edu

Yingyu Liang

University of Wisconsin-Madison

Madison, WI, USA

yliang@cs.wisc.edu

## Abstract

Fine-tuning (FT) pre-trained sentence embedding models on small datasets has been shown to have limitations. In this paper we show that concatenating the embeddings from the pre-trained model with those from a simple sentence embedding model trained only on the target data, can improve over the performance of FT for few-sample tasks. To this end, a linear classifier is trained on the combined embeddings, either by freezing the embedding model weights or training the classifier and embedding models end-to-end. We perform evaluation on seven small datasets from NLP tasks and show that our approach with end-to-end training outperforms FT with negligible computational overhead. Further, we also show that sophisticated combination techniques like CCA and KCCA do not work as well in practice as concatenation. We provide theoretical analysis to explain this empirical observation.

## 1 Introduction

Fine-tuning (FT) powerful pre-trained sentence embedding models like BERT (Devlin et al., 2018) has recently become the de-facto standard for downstream NLP tasks. Typically, FT entails jointly learning a classifier over the pre-trained model while tuning the weights of the latter. While FT has been shown to improve performance on tasks like GLUE (Wang et al., 2018) having large datasets (QQP, MNLI, QNLI), similar trends have not been observed on small datasets, where one would expect the maximum benefits of using a pre-trained model. Several works (Phang et al., 2018; Garg et al., 2019; Dodge et al., 2020; Lee et al., 2020) have demonstrated that FT with a few target domain samples is unstable with high variance, thereby often leading to sub-par gains. Furthermore, this

issue has also been well documented in practice <sup>1</sup>.

Learning with low resources has recently become an active research area in NLP, and arguably one of the most interesting scenarios for which pre-trained models are useful (e.g., (Cherry et al., 2019)). Many practical applications have small datasets (e.g., in social science, medical studies, etc), which are different from large-scale academic benchmarks having hundreds of thousands of training samples (e.g, DBpedia (Lehmann et al., 2015), Sogou News (Wang et al., 2008), etc). This necessitates effective transfer learning approaches using pre-trained sentence embedding models for few-sample tasks.

In this work, we show that concatenating sentence embeddings from a pre-trained model and those from a smaller model trained solely on the target data, can improve over the performance of FT. Specifically, we first learn a simple sentence embedding model on the target data. Then we concatenate( $C_{AT}$ ) the embeddings from this model with those from a pre-trained model, and train a linear classifier on the combined representation. The latter can be done by either freezing the embedding model weights or training the whole network (classifier plus the two embedding models) end-to-end.

We evaluate our approach on seven small datasets from NLP tasks. Our results show that our approach with end-to-end training can significantly improve the prediction performance of FT, with less than a 10% increase in the run time. Furthermore, our approach with frozen embedding models performs better than FT for very small datasets while reducing the run time by 30%–50%, and without the requirement of large memory GPUs.

We also conduct evaluations of multiple techniques for combining the pre-trained and domain-specific embeddings, comparing concatenation to

<sup>\*</sup>Equal contribution by authors

<sup>†</sup>Work completed at the University of Wisconsin-Madison

<sup>1</sup>Issues numbered 265, 1211 on <https://github.com/huggingface/transformers/issues/>

CCA and KCCA. We observe that the simplest approach of concatenation works best in practice. Moreover, we provide theoretical analysis to explain this empirical observation.

Finally, our results also have implications on the semantics learning ability of small domain-specific models compared to large pre-trained models. While intuition dictates that a large pre-trained model should capture the entire semantics learned by a small domain-specific model, our results show that there exist semantic features captured solely by the latter and not by the former, in spite of pre-training on billions of words. Hence combining the embeddings can improve the performance of directly FT the pre-trained model.

**Related Work** Recently, several pre-trained models have been studied, of which some provide explicit sentence embeddings (Conneau et al., 2017; Subramanian et al., 2018), while others provide implicit ones (Howard and Ruder, 2018; Radford et al., 2018). Peters et al. (2019) compare the performance of feature extraction (by freezing the pre-trained weights) and FT. There exists other more sophisticated transferring methods, but they are typically much more expensive or complicated. For example, Xu et al. (2019) “post-train” the pre-trained model on the target dataset, Houlsby et al. (2019) inject specifically designed new adapter layers, Arase and Tsujii (2019) inject phrasal paraphrase relations into BERT, Sun et al. (2019) use multi-task FT, and Wang et al. (2019) first train a deep network classifier on the fixed pre-trained embedding and then fine-tune it. Our focus is to propose alternatives to FT with similar simplicity and computational efficiency, and study conditions where it has significant advantages. While the idea of concatenating multiple embeddings has been previously used (Peters et al., 2018), we use it for transfer learning in a low resource target domain.

## 2 Methodology

We are given a set of labeled training sentences  $\mathcal{S}=\{(s_i, y_i)\}_{i=1}^m$  from a target domain and a pre-trained sentence embedding model  $f_1$ . Denote the embedding of  $s$  from  $f_1$  by  $v_{1s}=f_1(s)\in\mathbb{R}^{d_1}$ . Here  $f_1$  is assumed to be a large and powerful embedding model such as BERT. Our goal is to transfer  $f_1$  effectively to the target domain using  $\mathcal{S}$ . We propose to use a second sentence embedding model  $f_2$ , which is different from and typically much smaller than  $f_1$ , which has been trained solely on  $\mathcal{S}$ . The

small size of  $f_2$  is necessary for efficient learning on the small target dataset. Let  $v_{2s}=f_2(s)\in\mathbb{R}^{d_2}$  denote the embedding for  $s$  obtained from  $f_2$ .

Our method  $C_{AT}$  concatenates  $v_{1s}$  and  $v_{2s}$  to get an adaptive representation  $\bar{v}_s=[v_{1s}^\top, \alpha v_{2s}^\top]^\top$  for  $s$ . Here  $\alpha>0$  is a hyper-parameter to modify emphasis on  $v_{1s}$  and  $v_{2s}$ . It then trains a linear classifier  $c(\bar{v}_s)$  using  $\mathcal{S}$  in the following two ways:

**(a) Frozen Embedding Models** Only training the classifier  $c$  while fixing the weights of embedding models  $f_1$  and  $f_2$ . This approach is computationally cheaper than FT  $f_1$  since only  $c$  is trained. We denote this by  $C_{AT}\blacksquare$  (Locked  $f_1, f_2$  weights).

**(b) Trainable Embedding Models** Jointly training classifier  $c$ , and embedding models  $f_1, f_2$  in an end-to-end fashion. We refer to this as  $C_{AT}\blacksquare$ .

The inspiration for combining embeddings from two different models  $f_1, f_2$  stems from the impressive empirical gains of ensembling (Dietterich, 2000) in machine learning. While typical ensembling techniques like bagging and boosting aggregate predictions from individual models,  $C_{AT}\blacksquare$  and  $C_{AT}\blacksquare$  aggregate the embeddings from individual models and train a classifier using  $\mathcal{S}$  to get the predictions. Note that  $C_{AT}\blacksquare$  keeps the model weights of  $f_1, f_2$  frozen, while  $C_{AT}\blacksquare$  initializes the weights of  $f_2$  after initially training on  $\mathcal{S}$ <sup>2</sup>.

One of the benefits of  $C_{AT}\blacksquare$  and  $C_{AT}\blacksquare$  is that they treat  $f_1$  as a black box and do not access its internal architecture like other variants of FT (Houlsby et al., 2019). Additionally, we can theoretically guarantee that the concatenated embedding will generalize well to the target domain under assumptions on the loss function and embedding models.

## 2.1 Theoretical Analysis

Assume there exists a “ground-truth” embedding vector  $v_s^*$  for each sentence  $s$  with label  $y_s$ , and a “ground-truth” linear classifier  $f^*(s)=\langle w^*, v_s^* \rangle$  with a small loss  $L(f^*)=\mathbb{E}_s[\ell(f^*(s), y_s)]$  w.r.t. some loss function  $\ell$  (such as cross-entropy), where  $\mathbb{E}_s$  denotes the expectation over the true data distribution. The superior performance of  $C_{AT}\blacksquare$  in practice (see Section 3) suggests that there exists a linear relationship between the embeddings  $v_{1s}, v_{2s}$  and  $v_s^*$ . Thus we assume a theoretical model:  $v_{1s} = P_1 v_s^* + \epsilon_1$ ;  $v_{2s} = P_2 v_s^* + \epsilon_2$  where  $\epsilon_i$ ’s are noises independent of  $v_s^*$  with variances  $\sigma_i^2$ ’s. If we denote  $P^\top=[P_1^\top, P_2^\top]$  and  $\epsilon^\top=[\epsilon_1^\top, \epsilon_2^\top]$ , then

<sup>2</sup>We empirically observe that  $C_{AT}\blacksquare$  by randomly initializing weights of  $f_2$  performs similar to fine-tuning only  $f_1$

the concatenation  $\bar{v}_s = [v_{1s}^\top, v_{2s}^\top]^\top$  is  $\bar{v}_s = Pv_s^* + \epsilon$ . Let  $\sigma = \sqrt{\sigma_1^2 + \sigma_2^2}$ . We present the following theorem which guarantees the existence of a “good” classifier  $\bar{f}$  over  $\bar{v}_s$ :

**Theorem 1.** *If the loss function  $L$  is  $\lambda$ -Lipschitz for the first parameter, and  $P$  has full column rank, then there exists a linear classifier  $\bar{f}$  over  $\bar{v}_s$  such that  $L(\bar{f}) \leq L(f^*) + \lambda\sigma\|(P^\dagger)^\top w^*\|_2$  where  $P^\dagger$  is the pseudo-inverse of  $P$ .*

*Proof.* Let  $\bar{f}$  have weight  $\bar{w} = (P^\dagger)^\top w^*$ . Then

$$\begin{aligned}\langle \bar{w}, \bar{v}_s \rangle &= \langle (P^\dagger)^\top w^*, Pv_s^* + \epsilon \rangle \\ &= \langle (P^\dagger)^\top w^*, Pv_s^* \rangle + \langle (P^\dagger)^\top w^*, \epsilon \rangle \\ &= \langle w^*, P^\dagger Pv_s^* \rangle + \langle (P^\dagger)^\top w^*, \epsilon \rangle \\ &= \langle w^*, v_s^* \rangle + \langle (P^\dagger)^\top w^*, \epsilon \rangle.\end{aligned}\quad (1)$$

Then the difference in the losses is given by

$$\begin{aligned}L(\bar{f}) - L(f^*) &= \mathbb{E}_s[\ell(\bar{f}(s), y_s) - \ell(f^*(s), y_s)] \\ &\leq \lambda \mathbb{E}_s |\bar{f}(s) - f^*(s)|\end{aligned}\quad (2)$$

$$\begin{aligned}&= \lambda \mathbb{E}_s |\langle (P^\dagger)^\top w^*, \epsilon \rangle| \\ &\leq \lambda \sqrt{\mathbb{E}_s \langle (P^\dagger)^\top w^*, \epsilon \rangle^2}\end{aligned}\quad (3)$$

$$\begin{aligned}&\leq \lambda \sqrt{\mathbb{E}_s \|(P^\dagger)^\top w^*\|_2^2 \|\epsilon\|_2^2} \\ &= \lambda \sigma \|(P^\dagger)^\top w^*\|_2\end{aligned}\quad (4)$$

where we use the Lipschitz-ness of  $L$  in Equation 2, Jensen’s inequality in Equation 3, and Cauchy-Schwarz inequality in Equation 4.  $\square$

More intuitively, if the SVD of  $P = U\Sigma V^\top$ , then  $\|(P^\dagger)^\top w^*\|_2 = \|(\Sigma^\dagger)^\top V^\top w^*\|_2$ . So if the top right singular vectors in  $V$  align well with  $w^*$ , then  $\|(P^\dagger)^\top w^*\|_2$  will be small in magnitude. This means that if  $P_1$  and  $P_2$  together cover the direction  $w^*$ , they can capture information important for classification. And thus there exists a good classifier  $\bar{f}$  on  $\bar{v}_s$ . Additional explanation is presented in Appendix A.1.

## 2.2 Do Other Combination Methods Work?

There are several sophisticated techniques to combine  $v_{1s}$  and  $v_{2s}$  other than concatenation. Since  $v_{1s}$  and  $v_{2s}$  may be in different dimensions, a dimension reduction technique which projects them on the same dimensional space might work better at capturing the general and domain specific information. We consider two popular techniques:

**CCA** Canonical Correlation Analysis (Hotelling, 1936) learns linear projections  $\Phi_1$  and  $\Phi_2$  into dimension  $d$  to maximize the correlations between

the projections  $\{\Phi_1 v_{1s_i}\}$  and  $\{\Phi_2 v_{2s_i}\}$ . We use  $\bar{v}_s^\top = \frac{1}{2}\Phi_1 v_{1s_i} + \frac{1}{2}\Phi_2 v_{2s_i}$  with  $d = \min\{d_1, d_2\}$ .

**KCCA** Kernel Canonical Correlation Analysis (Schölkopf et al., 1998) first applies non-linear projections  $g_1$  and  $g_2$  and then CCA on  $\{g_1(v_{1s_i})\}_{i=1}^m$  and  $\{g_2(v_{2s_i})\}_{i=1}^m$ . We use  $d = \min\{d_1, d_2\}$  and  $\bar{v}_s^\top = \frac{1}{2}g_1(v_{1s_i}) + \frac{1}{2}g_2(v_{2s_i})$ .

We empirically evaluate  $C_{CA}$  and  $K_{CCA}$  and our results (see Section 3) show that the former two perform worse than  $C_{AT}$ . Further,  $C_{CA}$  performs even worse than the individual embedding models. This is a very interesting negative observation, and below we provide an explanation for this.

We argue that even when  $v_{1s}$  and  $v_{2s}$  contain information important for classification, CCA of the two embeddings can eliminate this and just retain the noise in the embeddings, thereby leading to inferior prediction performance. Theorem 2 constructs such an example.

**Theorem 2.** *Let  $\bar{v}_s$  denote the embedding for sentence  $s$  obtained by concatenation, and  $\tilde{v}_s$  denote that obtained by CCA. There exists a setting of the data and  $w^*, P, \epsilon$  such that there exists a linear classifier  $\bar{f}$  on  $\bar{v}_s$  with the same loss as  $f^*$ , while CCA achieves the maximum correlation but any classifier on  $\tilde{v}_s$  is at best random guessing.*

*Proof.* Suppose we perform CCA to get  $d$  dimensional  $\tilde{v}_s$ . Suppose  $v_s^*$  has  $d+2$  dimensions, each dimension being an independent Gaussian. Suppose  $w^* = [1, 1, 0, \dots, 0]^\top$ , and the label for the sentence  $s$  is  $y_s = 1$  if  $\langle w^*, v_s^* \rangle \geq 0$  and  $y_s = 0$  otherwise. Suppose  $\epsilon = 0$ ,  $P_1 = \text{diag}(1, 0, 1, \dots, 1)$ , and  $P_2 = \text{diag}(0, 1, 1, \dots, 1)$ .

Let the linear classifier  $\bar{f}$  have weights  $[1, 0, 0, 0, 1, \mathbf{0}]^\top$  where  $\mathbf{0}$  is the zero vector of  $d$  dimensions. Clearly,  $\bar{f}(s) = f^*(s)$  for any  $s$ , so it has the same loss as  $f^*$ .

For CCA, since the coordinates of  $v_s^*$  are independent Gaussians,  $v_{1s}$  and  $v_{2s}$  only have correlation in the last  $d$  dimensions. Solving the CCA optimization, the projection matrices for both embeddings are the same  $\phi = \text{diag}(0, 0, 1, \dots, 1)$  which achieves the maximum correlation. Then the CCA embedding is  $\tilde{v}_s = [0, 0, (v_s^*)_{3:(d+2)}]$  where  $(v_s^*)_{3:(d+2)}$  are the last  $d$  dimensions of  $v_s^*$ , which contains no information about the label. Therefore, any classifier on  $\tilde{v}_s$  is at best random guessing.  $\square$

The intuition for this is that  $v_{1s}$  and  $v_{2s}$  share com-

mon information while each has some special information for the classification. If the two sets of special information are uncorrelated, then they will be eliminated by CCA. Now, if the common information is irrelevant to the labels, then the best any classifier can do with the CCA embeddings is just random guessing. This is a fundamental drawback of the unsupervised CCA technique, clearly demonstrated by the extreme example in the theorem. In practice, the common information can contain some relevant information, so CCA embeddings are worse than concatenation but better than random guessing. KCCA can be viewed as CCA on a nonlinear transformation of  $v_{1s}$  and  $v_{2s}$  where the special information gets mixed non-linearly and cannot be separated out and eliminated by CCA. This explains why the poor performance of  $C_{\text{CA}} \blacksquare$  is not observed for  $K_{\text{CCA}} \blacksquare$  in Table 2. We present additional empirical verification of Theorem 2 in Appendix A.2.

### 3 Experiments

**Datasets** We evaluate our approach on seven low resource datasets from NLP text classification tasks like sentiment classification, question type classification, opinion polarity detection, subjectivity classification, etc. We group these datasets into 2 categories: the first having a few hundred training samples (which we term as very small datasets for the remainder of the paper), and the second having a few thousand training samples (which we term as small datasets). We consider the following 3 very small datasets: Amazon (product reviews), IMDB (movie reviews) and Yelp (food article reviews); and the following 4 small datasets: MR (movie reviews), MPQA (opinion polarity), TREC (question-type classification) and SUBJ (subjectivity classification). We present the statistics of the datasets in Table 1 and provide the details and downloadable links in Appendix B.1.

Dataset	c	N	V	Test
Amazon (Sarma et al., 2018)	2	1000	1865	100
IMDB (Sarma et al., 2018)	2	1000	3075	100
Yelp (Sarma et al., 2018)	2	1000	2049	100
MR (Pang and Lee, 2005)	2	10662	18765	1067
MPQA (Wiebe and Wilson, 2005)	2	10606	6246	1060
TREC (Li and Roth, 2002)	6	5952	9592	500
SUBJ (Pang and Lee, 2004)	2	10000	21323	1000

Table 1: Dataset statistics.  $c$ : Number of classes,  $N$ : Dataset size,  $|V|$ : Vocabulary size,  $\text{Test}$ : Test set size (if no standard test set is provided, we use a random train / dev / test split of 80 / 10 / 10 %)

	Amazon	Yelp	IMDB
BERT No-FT	93.1	90.2	91.6
BERT FT	94.0	91.7	92.3
Adapter	94.3	93.5	90.5
$C_{\text{NN-R}}$	91.1	92.7	93.2
$C_{\text{CA}} \blacksquare(C_{\text{NN-R}})$	79.1	71.5	80.8
$K_{\text{CCA}} \blacksquare(C_{\text{NN-R}})$	91.5	91.5	94.1
$C_{\text{AT}} \blacksquare(C_{\text{NN-R}})$	93.2	<b>96.5</b>	96.2
$C_{\text{AT}} \blacksquare(C_{\text{NN-R}})$	<b>94.0</b>	96.2	<b>97.0</b>
$C_{\text{NN-S}}$	94.7	95.2	96.6
$C_{\text{CA}} \blacksquare(C_{\text{NN-S}})$	83.6	67.8	83.3
$K_{\text{CCA}} \blacksquare(C_{\text{NN-S}})$	94.3	91.9	97.9
$C_{\text{AT}} \blacksquare(C_{\text{NN-S}})$	95.3	97.1	98.1
$C_{\text{AT}} \blacksquare(C_{\text{NN-S}})$	<b>95.7</b>	<b>97.2</b>	<b>98.3</b>
$C_{\text{NN-NS}}$	95.9	95.8	96.8
$C_{\text{CA}} \blacksquare(C_{\text{NN-NS}})$	81.3	69.4	85.0
$K_{\text{CCA}} \blacksquare(C_{\text{NN-NS}})$	95.8	96.2	97.2
$C_{\text{AT}} \blacksquare(C_{\text{NN-NS}})$	96.4	<b>98.3</b>	98.3
$C_{\text{AT}} \blacksquare(C_{\text{NN-NS}})$	<b>96.8</b>	<b>98.3</b>	<b>98.4</b>

Table 2: Evaluation on very small datasets.  $C_{\text{CA}} \blacksquare(\cdot)$  /  $K_{\text{CCA}} \blacksquare(\cdot)$  /  $C_{\text{AT}} \blacksquare(\cdot)$  /  $C_{\text{AT}} \blacksquare(\cdot)$  refers to using a specific CNN variant as  $f_2$ . Best results for each CNN variant in boldface.

**Models for Evaluation** We use the BERT (Devlin et al., 2018) base uncased model as the pre-trained model  $f_1$ . We choose a Text-CNN (Kim, 2014) model as the domain specific model  $f_2$  with 3 approaches to initialize the word embeddings: randomly initialized ( $C_{\text{NN-R}}$ ), static GloVe (Pennington et al., 2014) vectors ( $C_{\text{NN-S}}$ ) and trainable GloVe vectors ( $C_{\text{NN-NS}}$ ). We use a regularized logistic regression as the classifier  $c$ . We present the model and training details along with the chosen hyperparameters in Appendix B.2-B.3. We also present results with two other popular pre-trained models: GenSen and InferSent in Appendix C.2.

We consider two baselines: (i) BERT fine-tuning (denoted by BERT FT) and (ii) learning  $c$  over frozen pre-trained BERT weights (denoted by BERT No-FT). We also present the Adapter (Houlsby et al., 2019) approach as a baseline, which injects new adapters in BERT followed by selectively training the adapters while freezing the BERT weights, to compare with  $C_{\text{AT}} \blacksquare$  since neither fine-tunes the BERT parameters.

**Results on Very Small Datasets** On the 3 very small datasets, we present results averaged over 10 runs in Table 2. The key observations are summarized as follows:

(i)  $C_{\text{AT}} \blacksquare$  and  $C_{\text{AT}} \blacksquare$  almost always beat the accuracy of the baselines (BERT FT, Adapter) showing their effectiveness in transferring knowledge from the

	<b>MR</b>	<b>MPQA</b>	<b>SUBJ</b>	<b>TREC</b>
BERT No-FT	83.26	87.44	95.96	88.06
BERT FT	86.22	90.47	96.95	96.40
Adapter	85.55	90.40	97.40	96.55
C <sub>NN</sub> -NS	80.93	88.38	89.25	92.98
C <sub>AT</sub> (C <sub>NN</sub> -NS)	85.60	90.06	95.92	96.64
<b>C<sub>AT</sub> (C<sub>NN</sub>-NS)</b>	<b>87.15</b>	<b>91.19</b>	<b>97.60</b>	<b>97.06</b>

Table 3: Performance of C<sub>AT</sub> and C<sub>AT</sub> using C<sub>NN</sub>-NS and BERT on small datasets. Best results in boldface.

general domain to the target domain.

(ii) Both the C<sub>CA</sub>, K<sub>CCA</sub> (computationally expensive) get inferior performance than C<sub>AT</sub>. Similar trends for GenSen and InferSent in Appendix C.2. (iii) C<sub>AT</sub> performs better than C<sub>AT</sub>, but at an increased computational cost. The execution time for the latter is the time taken to train the text-CNN, extract BERT embeddings, concatenate them, and train a classifier on the combination. On an average run on the Amazon dataset, C<sub>AT</sub> requires about 125 s, reducing around 30% of the 180 s for BERT FT. Additionally, C<sub>AT</sub> has small memory requirements as it can be computed on a CPU in contrast to BERT FT which requires, at minimum, a 12GB memory GPU. The total time for C<sub>AT</sub> is 195 s, which is less than a 9% increase over FT. It also has a negligible 1.04% increase in memory (the number of parameters increases from 109,483,778 to 110,630,332 due to the text-CNN).

**Results on Small Datasets** We use the best performing C<sub>NN</sub>-NS model and present the results in Table 3. Again, C<sub>AT</sub> achieves the best performance on all the datasets improving the performance of BERT FT and Adapter. C<sub>AT</sub> can achieve comparable test accuracy to BERT FT on all the tasks while being much more computationally efficient. On an average run on the MR dataset, C<sub>AT</sub> (290 s) reduces the time of BERT FT (560 s) by about 50%, while C<sub>AT</sub> (610 s) only incurs an increase of about 9% over BERT FT.

**Comparison with Adapter** C<sub>AT</sub> can outperform Adapter for very small datasets and perform comparably on small datasets having 2 advantages:

- (i) We do not need to open the BERT model and access its parameters to introduce intermediate layers and hence our method is modular applicable to multiple pre-trained models.
- (ii) On very small datasets like Amazon, C<sub>AT</sub> introduces roughly only 1% extra parameters as compared to the 3–4% of Adapter thereby being more parameter efficient. However note that this increase

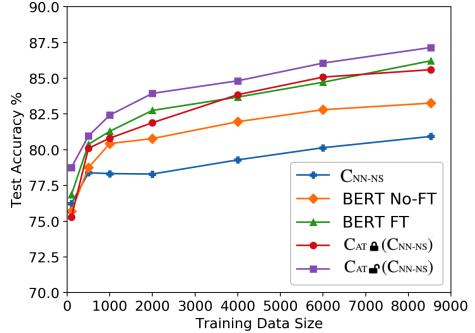


Figure 1: Comparing test accuracy of C<sub>AT</sub> and C<sub>AT</sub> on MR dataset with varying training dataset size.

in the number of parameters due to the text-CNN is a function of the vocabulary size of the dataset as it includes the word embeddings which are fed as input to the text-CNN. For a dataset having a larger vocabulary size like SUBJ<sup>3</sup>, Adapter might be more parameter efficient than C<sub>AT</sub>.

**Effect of Dataset Size** We study the effect of size of data on the performance of our method by varying the training data of the MR dataset via random sub-sampling. From Figure 1, we observe that C<sub>AT</sub> gets the best results across all training data sizes, significantly improving over BERT FT. C<sub>AT</sub> gets performance comparable to BERT FT on a wide range of data sizes, from 500 points on. We present qualitative analysis and complete results with error bounds in Appendix C.

## 4 Conclusion

In this paper we have proposed a simple method for transferring a pre-trained sentence embedding model for text classification tasks. We empirically show that concatenating pre-trained and domain specific sentence embeddings, learned on the target dataset, with or without fine-tuning can improve the classification performance of pre-trained models like BERT on small datasets. We have also provided theoretical analysis identifying the conditions when this method is successful and to explain the experimental results.

## Acknowledgements

This work was supported in part by FA9550-18-1-0166. The authors would also like to acknowledge the support provided by the University of Wisconsin-Madison Office of the Vice Chancellor for Research and Graduate Education with funding from the Wisconsin Alumni Research Foundation.

<sup>3</sup>For SUBJ, the embeddings alone contribute 6,396,900 additional parameters (5.84% of parameters of BERT-Base)

## References

- Yuki Arase and Jun’ichi Tsujii. 2019. Transfer fine-tuning: A BERT case study. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5393–5404, Hong Kong, China. Association for Computational Linguistics.
- Colin Cherry, Greg Durrett, George Foster, Reza Haf-fari, Shahram Khadivi, Nanyun Peng, Xiang Ren, and Swabha Swayamdipta, editors. 2019. *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*. Association for Computational Linguistics, Hong Kong, China.
- Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12(null):2493–2537.
- Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
- Thomas G. Dietterich. 2000. Ensemble methods in machine learning. In *Proceedings of the First International Workshop on Multiple Classifier Systems, MCS ’00*, page 1–15, Berlin, Heidelberg. Springer-Verlag.
- Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. 2020. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping.
- Siddhant Garg, Thuy Vu, and Alessandro Moschitti. 2019. Tanda: Transfer and adapt pre-trained transformer models for answer sentence selection.
- H Hotelling. 1936. Relations between two sets of variates. *Biometrika*.
- Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799.
- Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339.
- Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1746–1751.
- Cheolhyoung Lee, Kyunghyun Cho, and Wanmo Kang. 2020. Mixout: Effective regularization to finetune large-scale pretrained language models. In *International Conference on Learning Representations*.
- Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. 2015. DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web Journal*, 6(2).
- Xin Li and Dan Roth. 2002. Learning question classifiers. In *Proceedings of the 19th International Conference on Computational Linguistics - Volume 1, COLING ’02*, pages 1–7, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity. In *Proceedings of ACL*, pages 271–278.
- Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of ACL*, pages 115–124.
- Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.
- Matthew Peters, Sebastian Ruder, and Noah A Smith. 2019. To tune or not to tune? adapting pretrained representations to diverse tasks. *arXiv preprint arXiv:1903.05987*.
- Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of NAACL-HLT*, pages 2227–2237.
- Jason Phang, Thibault Févry, and Samuel R. Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks.
- Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. URL <https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/languageunsupervised/language-understanding-paper.pdf>.
- Prathusha K Sarma, Yingyu Liang, and Bill Sethares. 2018. Domain adapted word embeddings for improved sentiment classification. In *Proceedings of*

*the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 37–42.

Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. 1998. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319.

Sandeep Subramanian, Adam Trischler, Yoshua Bengio, and Christopher J Pal. 2018. Learning general purpose distributed sentence representations via large scale multi-task learning. *arXiv preprint arXiv:1804.00079*.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune BERT for text classification? *CoRR*, abs/1905.05583.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP*, Brussels, Belgium. Association for Computational Linguistics.

Canhui Wang, Min Zhang, Shaoping Ma, and Liyun Ru. 2008. Automatic online news issue construction in web environment. In *Proceedings of the 17th WWW*, page 457–466, New York, NY, USA. Association for Computing Machinery.

Ran Wang, Haibo Su, Chunye Wang, Kailin Ji, and Jupeng Ding. 2019. To tune or not to tune? how about the best of both worlds? *ArXiv*.

Janyce Wiebe and Theresa Wilson. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2):165–210.

Hu Xu, Bing Liu, Lei Shu, and S Yu Philip. 2019. Bert post-training for review reading comprehension and aspect-based sentiment analysis. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2324–2335.

## Appendix

### A Theorems: Additional Explanation

#### A.1 Concatenation

**Theorem 1.** *If the loss function  $L$  is  $\lambda$ -Lipschitz for the first parameter, and  $P$  has full column rank, then there exists a linear classifier  $\bar{f}$  over  $\bar{v}_s$  such that  $L(\bar{f}) \leq L(f^*) + \lambda\sigma\|(P^\dagger)^\top w^*\|_2$  where  $P^\dagger$  is the pseudo-inverse of  $P$ .*

**Justification of Assumptions** The assumption of Lipschitz-ness of the loss means that the loss changes smoothly with the prediction, which is a standard assumption in machine learning. The assumption on  $P$  having full column rank means that  $v_{1s}, v_{2s}$  contain the information of  $v_s^*$  and ensures that  $P^\dagger$  exists.<sup>4</sup>

**Explanation** For intuition about the term  $\|(P^\dagger)^\top w^*\|_2$ , consider the following simple example. Suppose  $v_s^*$  has 4 dimensions, and  $w^* = [1, 1, 0, 0]^\top$ , i.e., only the first two dimensions are useful for classification. Suppose  $P_1 = \text{diag}(c, 0, 1, 0)$  is a diagonal matrix, so that  $v_{1s}$  captures the first dimension with scaling factor  $c > 0$  and the third dimension with factor 1, and  $P_2 = \text{diag}(0, c, 0, 1)$  so that  $v_{2s}$  captures the other two dimensions. Hence we have  $(P^\dagger)^\top w^* = [1/c, 1/c, 0, 0]^\top$ , and thus

$$L(\bar{f}) \leq L(f^*) + \sqrt{2}\lambda\frac{\sigma}{c}$$

Thus the quality of the classifier is determined by the noise-signal ratio  $\sigma/c$ . If  $c$  is small, meaning that  $v_{1s}$  and  $v_{2s}$  mostly contain nuisance, then the loss is large. If  $c$  is large, meaning that  $v_{1s}$  and  $v_{2s}$  mostly capture the information along with some nuisance while the noise is relatively small, then the loss is close to that of  $f^*$ . Note that  $\bar{f}$  can be much better than any classifier using only  $v_{1s}$  or  $v_{2s}$  that has only part of the features determining the class labels.

#### A.2 CCA

**Theorem 2.** *Let  $\bar{v}_s$  denote the embedding for sentence  $s$  obtained by concatenation, and  $\tilde{v}_s$  denote that obtained by CCA. There exists a setting of the data and  $w^*, P, \epsilon$  such that there exists a linear classifier  $\bar{f}$  on  $\bar{v}_s$  with the same loss as  $f^*$ , while CCA achieves the maximum correlation but any classifier on  $\tilde{v}_s$  is at best random guessing.*

**Empirical Verification** One important insight from Theorem 2 is that when the two sets of embeddings have special information that is not shared with each other but is important for classification, then CCA will eliminate such information and have bad prediction performance. Let  $r_{2s} = v_{2s} - \Phi_2^\top \Phi_2 v_{2s}$  be the residue vector for the projection  $\Phi_2$  learned by CCA for the special domain, and similarly define  $r_{1s}$ . Then the analysis

---

<sup>4</sup>One can still do analysis dropping the full-rank assumption, but it will become more involved and non-intuitive

suggests that the residues  $r_{1s}$  and  $r_{2s}$  contain information important for prediction. We conduct experiments for BERT+CNN-non-static on Amazon reviews, and find that a classifier on the concatenation of  $r_{1s}$  and  $r_{2s}$  has accuracy 96.4%. This is much better than 81.3% on the combined embeddings via CCA. These observations provide positive support for our analysis.

## B Experiment Details

### B.1 Datasets

In addition to Table 1, here we provide details on the tasks of the datasets and links to download them for reproducibility of results.

- *Amazon*: A sentiment classification dataset on Amazon product reviews where reviews are classified as ‘Positive’ or ‘Negative’.<sup>5</sup>.
- *IMDB*: A sentiment classification dataset of movie reviews on IMDB where reviews are classified as ‘Positive’ or ‘Negative’<sup>3</sup>.
- *Yelp*: A sentiment classification dataset of restaurant reviews from Yelp where reviews are classified as ‘Positive’ or ‘Negative’<sup>3</sup>.
- *MR*: A sentiment classification dataset of movie reviews based on sentiment polarity and subjective rating (Pang and Lee, 2005)<sup>6</sup>.
- *MPQA*: An unbalanced polarity classification dataset ( 70% negative examples) for opinion polarity detection (Wiebe and Wilson, 2005)<sup>7</sup>.
- *TREC*: A question type classification dataset with 6 classes for questions about a person, location, numeric information, etc. (Li and Roth, 2002)<sup>8</sup>.
- *SUBJ*: A dataset for classifying a sentence as having subjective or objective opinions (Pang and Lee, 2004).

The Amazon, Yelp and IMDB review datasets have previously been used for research on few-sample learning by Sarma et al. (2018) and capture sentiment information from target domains very different from the general text corpora of the pre-trained models.

<sup>5</sup><https://archive.ics.uci.edu/ml/datasets/Sentiment+Labelled+Sentences>

<sup>6</sup><https://www.cs.cornell.edu/people/pabo/movie-review-data/>

<sup>7</sup><http://mpqa.cs.pitt.edu/>

<sup>8</sup><http://cogcomp.org/Data/QA/QC/>

### B.2 Embedding Models

#### B.2.1 Domain Specific $f_2$

We use the text-CNN model (Kim, 2014) for domain specific embeddings  $f_2$  the details of which are provided below.

**Text-CNN** The model restricts the maximum sequence length of the input sentence to 128 tokens, and uses convolutional filter windows of sizes 3, 4, 5 with 100 feature maps for each size. A max-overtime pooling operation (Collobert et al., 2011) is used over the feature maps to get a 384 dimensional sentence embeddings (128 dimensions corresponding to each filter size). We train the model using the Cross Entropy loss with an  $\ell_2$  norm penalty on the classifier weights similar to Kim (2014). We use a dropout rate of 0.5 while training. For each dataset, we create a vocabulary specific to the dataset which includes any token present in the train/dev/test split. The input word embeddings can be chosen in the following three ways:

- **$C_{NN-R}$**  : Randomly initialized 300-dimensional word embeddings trained together with the text-CNN.
- **$C_{NN-S}$**  : Initialised with GloVe (Pennington et al., 2014) pre-trained word embeddings and made static during training the text-CNN.
- **$C_{NN-NS}$**  : Initialised with GloVe (Pennington et al., 2014) pre-trained word embeddings and made trainable during training the text-CNN.

For very small datasets we additionally compare with sentence embeddings obtained using the Bag of Words approach.

#### B.2.2 Pre-Trained $f_1$

We use the following three models for pre-trained embeddings  $f_1$ :

**BERT** We use the BERT<sup>9</sup>-base uncased model with WordPiece tokenizer having 12 transformer layers. We obtain 768 dimensional sentence embeddings corresponding to the [CLS] token from the final layer. We perform fine-tuning for 20 epochs with early stopping by choosing the best performing model on the validation data. The additional fine-tuning epochs (20 compared to the typical 3) allows for a better performance of the fine-tuning baseline since we use early stopping.

<sup>9</sup><https://github.com/google-research/bert>

**InferSent** We use the pre-trained InferSent model (Conneau et al., 2017) to obtain 4096 dimensional sentence embeddings using the implementation provided in the SentEval<sup>10</sup> repository. We use InferSent v1 for all our experiments.

**GenSen** We use the pre-trained GenSen model (Subramanian et al., 2018) implemented in the SentEval repository to obtain 4096 dimensional sentence embeddings.

### B.3 Training Details

We train domain specific embeddings on the training data and extract the embeddings. We combine these with the embeddings from the pre-trained models and train a regularized logistic regression classifier on top. This classifier is learned on the training data, while using the dev data for hyper-parameter tuning the regularizer penalty on the weights. The classifier can be trained either by freezing the weights of the embedding models or training the whole network end-to-end. The performance is tested on the test set. We use test accuracy as the performance metric and report all results averaged over 10 experiments unless mentioned otherwise. The experiments are performed on an NVIDIA Titan Xp 12 GB GPU.

#### B.3.1 Hyperparameters

We use an Adam optimizer with a learning rate of  $2e^{-5}$  as per the standard fine-tuning practice. For  $C_{CA} \&$ , we used a regularized CCA implementation and tune the regularization parameter via grid search in  $[0.00001, 10]$  in multiplicative steps of 10 over the validation data. For  $K_{CCA} \&$ , we use a Gaussian kernel with a regularized KCCA implementation where the Gaussian sigma and the regularization parameter are tuned via grid search in  $[0.05, 10]$  and  $[0.00001, 10]$  respectively in multiplicative steps of 10 over the validation data. For  $C_{AT} \&$  and  $C_{AT} \&$ , the weighting parameter  $\alpha$  is tuned via grid search in the range  $[0.002, 500]$  in multiplicative steps of 10 over the validation data.

## C Additional Results

### C.1 Qualitative Analysis

We present some qualitative examples from the Amazon, IMDB and Yelp datasets on which BERT and  $C_{NN-NS}$  are unable to provide the correct class predictions, while  $C_{AT} \&$  or  $K_{CCA} \&$  can successfully provide the correct class predictions in Table 4.

<sup>10</sup><https://github.com/facebookresearch/SentEval>

#### Correctly classified by $K_{CCA} \&$

However-the ringtones are not the best, and neither are the games.

This is cool because most cases are just open there allowing the screen to get all scratched up.

#### Correctly classified by $C_{AT} \&$

TNot nearly as good looking as the amazon picture makes it look .

Magical Help .

(a) Amazon

#### Correctly classified by $K_{CCA} \&$

I would have casted her in that role after ready the script . Predictable , but not a bad watch .

#### Correctly classified by $C_{AT} \&$

I would have casted her in that role after ready the script . Predictable , but not a bad watch .

(b) IMDB

#### Correctly classified by $K_{CCA} \&$

The lighting is just dark enough to set the mood .

I went to Bachi Burger on a friend’s recommendation and was not disappointed . dont go here .

I found this place by accident and I could not be happier .

#### Correctly classified by $C_{AT} \&$

The lighting is just dark enough to set the mood .

I went to Bachi Burger on a friend’s recommendation and was not disappointed . dont go here .

I found this place by accident and I could not be happier .

(c) Yelp

Table 4: Sentences from Amazon, IMDB, Yelp datasets where  $K_{CCA} \&$  and  $C_{AT} \&$  of BERT and  $C_{NN-NS}$  embeddings succeeds while they individually give wrong predictions.

We observe that these are either short sentences or ones where the content is tied to the specific reviewing context as well as the involved structure to be parsed with general knowledge. Such input sentences thus require combining both the general semantics of BERT and the domain specific semantics of  $C_{NN-NS}$  to predict the correct class labels.

## C.2 Complete Results with Error Bounds

We present a comprehensive set of results along with error bounds on very small datasets (Amazon, IMDB and Yelp reviews) in Table 2, where we evaluate three popularly used pre-trained sentence embedding models, namely BERT, GenSen and InferSent. We present the error bounds on the results for small datasets in Table 3. For small datasets, we additionally present results from using  $C_{CA} \&$  (We omit  $K_{CCA} \&$  here due to high computational memory requirements).

			<b>BOW</b>	<b>C<sub>NN-R</sub></b>	<b>C<sub>NN-S</sub></b>	<b>C<sub>NN-NS</sub></b>	
Amazon	<b>Default</b>			79.20 $\pm$ 2.31	91.10 $\pm$ 1.64	94.70 $\pm$ 0.64	95.90 $\pm$ 0.70
	<b>BERT</b>	94.00 $\pm$ 0.02	C <sub>AT-#</sub>	-	94.05 $\pm$ 0.23	95.70 $\pm$ 0.50	<b>96.75 <math>\pm</math> 0.76</b>
			C <sub>AT-#</sub>	89.59 $\pm$ 1.22	93.20 $\pm$ 0.98	95.30 $\pm$ 0.46	96.40 $\pm$ 1.11
			KCCA <sub>#</sub>	89.12 $\pm$ 0.47	91.50 $\pm$ 1.63	94.30 $\pm$ 0.46	95.80 $\pm$ 0.40
			C <sub>CA-#</sub>	50.91 $\pm$ 1.12	79.10 $\pm$ 2.51	83.60 $\pm$ 1.69	81.30 $\pm$ 3.16
	<b>GenSen</b>	82.55 $\pm$ 0.82	C <sub>AT-#</sub>	82.82 $\pm$ 0.97	92.80 $\pm$ 1.25	94.10 $\pm$ 0.70	95.00 $\pm$ 1.0
			KCCA <sub>#</sub>	79.21 $\pm$ 2.28	91.30 $\pm$ 1.42	94.80 $\pm$ 0.75	<b>95.90 <math>\pm</math> 0.30</b>
			C <sub>CA-#</sub>	52.80 $\pm$ 0.74	80.60 $\pm$ 4.87	83.00 $\pm$ 2.45	84.95 $\pm$ 1.45
	<b>InferSent</b>	85.29 $\pm$ 1.61	C <sub>AT-#</sub>	51.89 $\pm$ 0.62	90.30 $\pm$ 1.48	94.70 $\pm$ 1.10	95.90 $\pm$ 0.70
			KCCA <sub>#</sub>	52.29 $\pm$ 0.74	91.70 $\pm$ 1.49	95.00 $\pm$ 0.00	<b>96.00 <math>\pm</math> 0.00</b>
			C <sub>CA-#</sub>	53.10 $\pm$ 0.82	61.10 $\pm$ 3.47	65.50 $\pm$ 3.69	71.40 $\pm$ 3.04
Yelp	<b>Default</b>			81.3 $\pm$ 2.72	92.71 $\pm$ 0.46	95.25 $\pm$ 0.39	95.83 $\pm$ 0.14
	<b>BERT</b>	91.67 $\pm$ 0.00	C <sub>AT-#</sub>	-	96.23 $\pm$ 1.04	97.23 $\pm$ 0.70	<b>98.34 <math>\pm</math> 0.62</b>
			C <sub>AT-#</sub>	89.03 $\pm$ 0.70	96.50 $\pm$ 1.33	97.10 $\pm$ 0.70	98.30 $\pm$ 0.78
			KCCA <sub>#</sub>	88.51 $\pm$ 1.22	91.54 $\pm$ 4.63	91.91 $\pm$ 1.13	96.2 $\pm$ 0.87
			C <sub>CA-#</sub>	50.27 $\pm$ 1.33	71.53 $\pm$ 2.46	67.83 $\pm$ 3.07	69.4 $\pm$ 3.35
	<b>GenSen</b>	86.75 $\pm$ 0.79	C <sub>AT-#</sub>	85.94 $\pm$ 1.04	94.24 $\pm$ 0.53	95.77 $\pm$ 0.36	<b>96.03 <math>\pm</math> 0.23</b>
			KCCA <sub>#</sub>	83.35 $\pm$ 1.79	92.58 $\pm$ 0.31	95.41 $\pm$ 0.45	95.06 $\pm$ 0.56
			C <sub>CA-#</sub>	57.14 $\pm$ 0.84	84.27 $\pm$ 1.68	86.94 $\pm$ 1.62	87.27 $\pm$ 1.81
	<b>InferSent</b>	85.7 $\pm$ 1.12	C <sub>AT-#</sub>	50.83 $\pm$ 0.42	91.94 $\pm$ 0.46	96.10 $\pm$ 1.30	<b>97.00 <math>\pm</math> 0.77</b>
			KCCA <sub>#</sub>	50.80 $\pm$ 0.65	91.13 $\pm$ 1.63	95.45 $\pm$ 0.23	95.57 $\pm$ 0.55
			C <sub>CA-#</sub>	55.91 $\pm$ 1.23	60.80 $\pm$ 2.22	54.70 $\pm$ 1.34	59.50 $\pm$ 1.85
IMDB	<b>Default</b>			89.30 $\pm$ 1.00	93.25 $\pm$ 0.38	96.62 $\pm$ 0.46	96.76 $\pm$ 0.26
	<b>BERT</b>	92.33 $\pm$ 0.00	C <sub>AT-#</sub>	-	97.07 $\pm$ 0.95	98.31 $\pm$ 0.83	<b>98.42 <math>\pm</math> 0.78</b>
			C <sub>AT-#</sub>	89.27 $\pm$ 0.97	96.20 $\pm$ 2.18	98.10 $\pm$ 0.94	98.30 $\pm$ 1.35
			KCCA <sub>#</sub>	88.29 $\pm$ 0.65	94.10 $\pm$ 1.87	97.90 $\pm$ 0.30	97.20 $\pm$ 0.40
			C <sub>CA-#</sub>	51.03 $\pm$ 1.20	80.80 $\pm$ 2.75	83.30 $\pm$ 4.47	84.97 $\pm$ 1.44
	<b>GenSen</b>	86.41 $\pm$ 0.66	C <sub>AT-#</sub>	86.86 $\pm$ 0.62	95.63 $\pm$ 0.47	97.22 $\pm$ 0.27	<b>97.42 <math>\pm</math> 0.31</b>
			KCCA <sub>#</sub>	84.72 $\pm$ 0.93	93.23 $\pm$ 0.38	96.19 $\pm$ 0.21	96.60 $\pm$ 0.37
			C <sub>CA-#</sub>	51.48 $\pm$ 1.02	86.28 $\pm$ 1.76	87.30 $\pm$ 2.12	87.47 $\pm$ 2.17
	<b>InferSent</b>	84.3 $\pm$ 0.63	C <sub>AT-#</sub>	50.36 $\pm$ 0.62	92.30 $\pm$ 1.26	97.90 $\pm$ 1.37	97.10 $\pm$ 1.22
			KCCA <sub>#</sub>	50.09 $\pm$ 0.68	92.40 $\pm$ 1.11	97.62 $\pm$ 0.48	<b>98.20 <math>\pm</math> 1.40</b>
			C <sub>CA-#</sub>	52.56 $\pm$ 1.15	54.50 $\pm$ 4.92	54.20 $\pm$ 5.15	61.00 $\pm$ 4.64

Table 5: Test accuracy ( $\pm$  std dev) for Amazon, Yelp and IMDB review datasets. Default values are performance of the domain specific models. Default values for BERT, Gensen and InferSent correspond to fine-tuning them. Best results for each pre-trained model are highlighted in boldface.

	<b>MR</b>	<b>MPQA</b>	<b>SUBJ</b>	<b>TREC</b>
<b>BERT No-FT</b>	83.26 $\pm$ 0.67	87.44 $\pm$ 1.37	95.96 $\pm$ 0.27	88.06 $\pm$ 1.90
<b>BERT FT</b>	86.22 $\pm$ 0.85	90.47 $\pm$ 1.04	96.95 $\pm$ 0.14	96.40 $\pm$ 0.67
<b>C<sub>NN-NS</sub></b>	80.93 $\pm$ 0.16	88.38 $\pm$ 0.28	89.25 $\pm$ 0.08	92.98 $\pm$ 0.89
<b>C<sub>CA-#</sub>(C<sub>NN-NS</sub>)</b>	85.41 $\pm$ 1.18	77.22 $\pm$ 1.82	94.55 $\pm$ 0.44	84.28 $\pm$ 2.96
<b>C<sub>AT-#</sub>(C<sub>NN-NS</sub>)</b>	85.60 $\pm$ 0.95	90.06 $\pm$ 0.48	95.92 $\pm$ 0.26	96.64 $\pm$ 1.07
<b>C<sub>AT-#</sub>(C<sub>NN-NS</sub>)</b>	<b>87.15 <math>\pm</math> 0.70</b>	<b>91.19 <math>\pm</math> 0.84</b>	<b>97.60 <math>\pm</math> 0.23</b>	<b>97.06 <math>\pm</math> 0.48</b>

Table 6: Test accuracy ( $\pm$  std dev) for MR, MPQA, SUBJ and TREC datasets. Best results on the datasets are highlighted in boldface. The domain specific embedding model used is CNN-non-static, and the pre-trained model used is BERT.

# Multimodal Pretraining for Dense Video Captioning

Gabriel Huang<sup>1\*</sup>, Bo Pang<sup>2</sup>, Zhenhai Zhu<sup>2</sup>, Clara Rivera<sup>2</sup>, Radu Soricut<sup>2</sup>

<sup>1</sup>Mila & University of Montreal

<sup>2</sup>Google Research

gabriel.huang@umontreal.ca

{bopang,zhenhai,rivera,rsoricut}@google.com

## Abstract

Learning specific hands-on skills such as cooking, car maintenance, and home repairs increasingly happens via instructional videos. The user experience with such videos is known to be improved by meta-information such as time-stamped annotations for the main steps involved. Generating such annotations automatically is challenging, and we describe here two relevant contributions. First, we construct and release a new dense video captioning dataset, Video Timeline Tags (ViTT), featuring a variety of instructional videos together with time-stamped annotations. Second, we explore several multimodal sequence-to-sequence pretraining strategies that leverage large unsupervised datasets of videos and caption-like texts. We pretrain and subsequently finetune dense video captioning models using both YouCook2 and ViTT. We show that such models generalize well and are robust over a wide variety of instructional videos.

## 1 Introduction

YouTube recently reported that a billion hours of videos were being watched on the platform every day ([YouTubeBlog, 2017](#)). In addition, the amount of time people spent watching online videos was estimated to grow at an average rate of 32% a year between 2013 and 2018, with an average person forecasted to watch 100 minutes of online videos per day in 2021 ([ZenithMedia, 2019](#)).

An important reason for this fast-growing video consumption is information-seeking. For instance, people turn to YouTube “hungry for how-to and learning content” ([O’Neil-Hart, 2018](#)). Indeed, compared to traditional content format such as text, video carries richer information to satisfy such

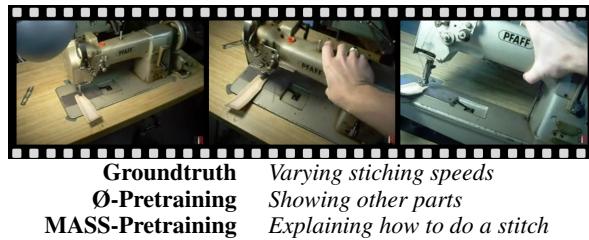


Figure 1: Dense video captioning using ViTT-trained models. For the given video scene, we show the ViTT annotation (Groundtruth) and model outputs (no pre-training and MASS-based pretraining).

needs. But as a content media, videos are also inherently more difficult to skim through, making it harder to quickly target the relevant part(s) of a video. Recognizing this difficulty, search engines started showing links to “key moments” within videos in search results, based on timestamps and short descriptions provided by the content creators themselves.<sup>1</sup> This enables users to get a quick sense of what the video covers, and also to jump to a particular time in the video if so desired. This effort echoes prior work in the literature showing how users of instructional videos can benefit from human-curated meta-data, such as a timeline pointing to the successive steps of a tutorial ([Kim et al., 2014](#); [Margulieux et al., 2012](#); [Weir et al., 2015](#)). Producing such meta-data in an automatic way would greatly scale up the efforts of providing easier information access to videos. This task is closely related to the dense video captioning task considered in prior work ([Zhou et al., 2018a,c](#); [Krishna et al., 2017](#)), where an instructional video is first segmented into its main steps, followed by segment-level caption generation.

To date, the YouCook2 data set ([Zhou et al., 2018a](#)) is the largest annotated data set for dense

\*This work was done while Gabriel Huang was an intern at Google Research.

<sup>1</sup><https://www.blog.google/products/search/key-moments-video-search/>

video captioning. It contains annotations for 2,000 cooking videos covering 89 recipes, with per-recipe training / validation split. Restricting to a small number of recipes is helpful for early exploratory work, but such restrictions impose barriers to model generalization and adoption that are hard to overcome. We directly address this problem by constructing a larger and broader-coverage annotated dataset that covers a wide range of instructional topics (cooking, repairs, maintenance, etc.) We make the results of our annotation efforts publicly available as Video Timeline Tags (ViTT)<sup>2</sup>, consisting of around 8,000 videos annotated with timelines (on average 7.1 segments per video, each segment with a short free-text description).

Using YouCook2 and the new ViTT dataset as benchmarks for testing model performance and generalization, we further focus on the subproblem of video-segment-level caption generation, assuming segment boundaries are given (Hessel et al., 2019; Sun et al., 2019b; Luo et al., 2020). Motivated by the high cost of collecting human annotations, we investigate pretraining a video segment captioning model using unsupervised signals – ASR (Automatic Speech Recognition) tokens and visual features from instructional videos, and *unpaired* instruction steps extracted from independent sources: Recipe1M (Marin et al., 2019) and WikiHow (Koupaee and Wang, 2018). In contrast to prior work that focused on BERT-style pretraining of encoder networks (Sun et al., 2019b,a), our approach entails jointly pretraining both multimodal encoder and text-based decoder models via MASS-style pretraining (Song et al., 2019). Our experiments show that pretraining with either text-only or multi-modal data provides significant gains over no pretraining, on both the established YouCook2 benchmark and the new ViTT benchmark. The results we obtain establish state-of-the-art performance on YouCook2, and present strong performance numbers on the ViTT benchmark. These findings help us conclude that the resulting models generalize well and are quite robust over a wide variety of instructional videos.

## 2 Related Work

**Text-only Pretraining.** Language pretraining models based on the Transformer neural net-

work architecture (Vaswani et al., 2017a) such as BERT (Devlin et al., 2018), GPT (Radford et al., 2018), RoBERTa (Liu et al., 2019), MASS (Song et al., 2019) and ALBERT (Lan et al., 2020) have achieved state-of-the-art results on many NLP tasks. MASS (Song et al., 2019) has been recently proposed as a joint encoder-decoder pretraining strategy. For sequence-to-sequence tasks, this strategy is shown to outperform approaches that separately pretrain the encoder (using a BERT-style objective) and the decoder (using a language modeling objective). UniLM (Dong et al., 2019), BART (Lewis et al., 2019), and T5 (Raffel et al., 2019) propose unified pretraining approaches for both understanding and generation tasks.

**Multimodal Pretraining.** VideoBERT (Sun et al., 2019b), CBT (Sun et al., 2019a) and ActBERT (Zhu and Yang, 2020) use a BERT-style objective to train both video and ASR text encoders. Alayrac et al. (2016) and Miech et al. (2020) use margin-based loss functions to learn joint representations for video and ASR, and evaluate them on downstream tasks such as video captioning, action segmentation and anticipation, and action localization. An independent and concurrent work (UniViLM) by Luo et al. (2020) is closely related to ours in that we share some similar pretraining objectives, some of the pretraining setup – HowTo100M (Alayrac et al., 2016), and the down-stream video captioning benchmark using YouCook2 (Zhou et al., 2018a). The main difference is that they use BERT-style pretraining for encoder and language-modeling style pretraining for decoder, whereas we use MASS-style pre-training to pretrain encoder and decoder jointly.

Other approaches such as ViLBERT (Lu et al., 2019), LXMERT (Tan and Bansal, 2019), Unicoder-VL (Li et al., 2019), VL-BERT (Su et al., 2019), and UNITER (Chen et al., 2019) focus on pretraining joint representations for text and image, evaluating them on downstream tasks such as visual question answering, image-text retrieval and referring expressions.

**Dense Video Captioning.** In this paper, we focus on generating captions at the segment-level, which is a sub-task of the so-called dense video captioning task (Krishna et al., 2017), where fine-grained captions are generated for video segments, conditioned on an input video with pre-defined

---

<sup>2</sup>Available at <https://github.com/google-research-datasets/Video-Timeline-Tags-ViTT>

Name	Type	# segments
<i>Pretraining datasets</i>		
YT8M-cook	ASR+video	186 K
HowTo100M	ASR+video	8.0 M
Recipe1M	CAP-style	10.8 M
WikiHow	CAP-style	1.3 M
<i>Finetuning datasets</i>		
YouCook2	ASR+video+CAP	11.5 K
ViTT-All	ASR+video+CAP	88.5 K

Table 1: Datasets used in this work, along with size of the data measured by the total number of segments.

event segments. This is different from the video captioning models that generate a single summary for the entire video (Wang et al., 2019).

Hessel et al. (2019) make use of ASR and video for segment-level captioning on YouCook2 and show that most of the performance comes from ASR. Shi et al. (2019); Luo et al. (2020) train their dense video captioning models on both video frames and ASR text and demonstrate the benefits of adding ASR as an input to the model. There are also a number of video captioning approaches that do not use ASR directly (Zhou et al., 2018c; Pan et al., 2020; Zheng et al., 2020; Zhang et al., 2020; Lei et al., 2020).

**Instructional video captioning data sets.** In addition to YouCook2 (Zhou et al., 2018a), there are two other smaller data sets in the instructional video captioning category. Epic Kitchen (Damen et al., 2018) features 55 hours of video consisting of 11.5M frames, which were densely labeled for a total of 39.6K action segments and 454.3K object bounding boxes. How2 (Sanabria et al., 2018) consists of instructional videos with video-level (as opposed to segment-level) descriptions, authored by the video creators themselves.

### 3 Data

We present the datasets used for pretraining, fine-tuning, and evaluation in Table 1. We also describe in detail the newly introduced dense video captioning dataset, Video Timeline Tags (ViTT).

#### 3.1 Dense Video-Captioning Datasets

Our goal is to generate captions (CAP) for video segments. We consider two datasets with segment-level captions for fine-tuning and evaluating ASR+Video→CAP models.

**YouCook2.** Up to this point, YouCook2 (Zhou et al., 2018a) has been the largest human-annotated dense-captioning dataset of instructional videos publicly available. It originally contained 2,000 cooking videos from YouTube. Starting from 110 recipe types (e.g., “shrimp tempura”), 25 unique videos per recipe type were collected; the recipe types that did not gather enough videos were dropped, resulting in a total of 89 recipe types in the final dataset. In addition, Zhou et al. (2018b) “randomly split the videos belonging to each recipe into 67%:23%:10% as training, validation and test sets<sup>3</sup>,” which effectively guarantees that videos in the validation and test sets are never about unseen recipes. Annotators were then asked to construct recipe steps for each video — that is, identify the start and end times for each step, and provide a recipe-like description of each step. Overall, they reported an average of 7.7 segments per video, and 8.8 words per description. After removing videos that had been deleted by users, we obtained a total of 11,549 segments.

**ViTT.** One limitation of the YouCook2 dataset is the artificially imposed (almost) uniform distribution of videos over 89 recipes. While this may help making the task more tractable, it is difficult to judge whether performance on its validation / test sets can be generalized to unseen topics.

The design of our ViTT dataset annotation process is aimed at fixing some of these drawbacks. We started by collecting a large dataset of videos containing a broader variety of topics to better reflect topic distribution in the wild. Specifically, we randomly sampled instructional videos from the YouTube-8M dataset (Abu-El-Haija et al., 2016), a large-scale collection of YouTube videos that also contain topical labels. Since much of prior work in this area revolved around cooking videos, we aimed at sampling a significant proportion of our data from videos with cooking labels (specifically, “Cooking” and “Recipe”). Aside from the intentional bias regarding cooking videos, the rest of the videos were selected by randomly sampling non-cooking videos, including only those that were considered to be instructional videos by our human annotators.

Once candidate videos were identified, timeline annotations and descriptive tags were collected.

---

<sup>3</sup>Note that no annotations are provided for the test split; we conducted our own training/dev/test split over available videos.

Our motivation was to enable downstream applications to allow navigating to specific content sections. Therefore, annotators were asked to identify the main steps in a video and mark their start time. They were also asked to produce a descriptive-yet-concise, free-text tag for each step (e.g., “shaping the cookies”, “removing any leftover glass”). A subset of the videos has received more than one complete annotation (main steps plus tags).

The resulting ViTT dataset consists of a total of 8,169 videos, of which 3,381 are cooking-related. A total of 5,840 videos have received only one annotation, and have been designated as the training split. Videos with more than one annotation have been designated as validation / test data. Overall, there are 7.1 segments per video on average (max: 19). Given the dataset design, descriptions are much shorter in length compared to YouCook2: on average there are 2.97 words per tag (max: 16) — 20% of the captions are single-word, 22% are two-words, and 25% are three words. Note that the average caption length is significantly shorter than for YouCook2, which is not surprising given our motivation of providing short and concise timeline tags for video navigation. We standardized the paraphrases among the top-20 most frequent captions. For instance, {“intro”, “introduction”} → “intro”. Otherwise, we have preserved the original tags as-is, even though additional paraphrasing most definitely exists. Annotators were instructed to start and end the video with an opening and closing segment as possible. As a result, the most frequent tag (post-standardization) in the dataset is “intro”, which accounts for roughly 11% of the 88,455 segments. More details on the data collection process and additional analysis can be found in the Supplementary Material (Section A.1).

Overall, this results in 56,027 unique tags, with a vocabulary size of 12,509 token types over 88,455 segments. In this paper, we consider two variants: the full dataset (ViTT-All), and the cooking subset (ViTT-Cooking).

### 3.2 Pretraining Datasets: ASR+Video

We consider two large-scale unannotated video datasets for pretraining, as described below. Time-stamped ASR tokens were obtained via YouTube Data API<sup>4</sup> and split into ASR segments if the timestamps of two consecutive words are more

than 2 seconds apart, or if a segment is longer than a pre-specified max length (in our case, 320 words). They were paired with concurrent video frames in the same segment.

**YT8M-cook** We extract the cooking subset of YouTube-8M (Abu-El-Haija et al., 2016) by taking, from its training split, videos with “Cooking” or “Recipe” labels, and retain those with English ASR, subject to YouTube policies. After preprocessing, we obtain 186K ASR+video segments with an average length of 64 words (24 seconds) per segment.

**HowTo100M.** This is based on the 1.2M YouTube instructional videos released by Miech et al. (2019), covering a broad range of topics. After preprocessing, we obtain 7.99M ASR+video segments with an average of 78 words (28.7 seconds) per segment.

### 3.3 Pretraining Datasets: CAP-style

We also consider two text-only datasets for *pre-training*, containing *unpaired* instruction steps similar in style to the target captions.

**Recipe1M** is a collection of 1M recipes scraped from a number of popular cooking websites (Marin et al., 2019). We use the sequence of instructions extracted for each recipe in this dataset, and treat each recipe step as a separate example during pre-training. This results in 10,767,594 CAP-style segments, with 12.8 words per segment.

**WikiHow** is a collection of 230,843 articles extracted from the WikiHow knowledge base (Koupaee and Wang, 2018). Each article comes with a title starting with “How to”. Each associated step starts with a step summary (in bold) followed by a detailed explanation. We extract the all step summaries, resulting in 1,360,145 CAP-style segments, with 8.2 words per segment. Again, each instruction step is considered as a separate example during pretraining.

### 3.4 Differences between Pretraining and Finetuning Datasets

First, note that *video segments* are defined differently for pretraining and finetuning datasets, and may not match exactly. For ASR+Video pretraining datasets, which are unsupervised, the segments are divided following a simple heuristic (e.g., two consecutive words more than 2 seconds apart), whereas for finetuning ASR+Video→CAP datasets, which are supervised, the segments are defined by

<sup>4</sup><https://developers.google.com/youtube/v3/docs/captions>

human annotators to correspond to instruction steps. Otherwise, the ASR data are relatively similar between pretraining and finetuning datasets, as both come from instructional videos and are in the style of spoken language.

Second, compared to the target captions in finetuning datasets, the CAP-like pretraining datasets are similar in spirit — they all represent summaries of *steps*, but they may differ in length, style and granularity. In particular, the CAP-like pretraining datasets are closer in style to captions in YouCook2, where annotators were instructed to produce a recipe-like description for each step. This is reflected in their similar average length (YouCook2: 8.8 words, Recipe1M: 12.8 words, WikiHow: 8.2 words); whereas captions in ViTT are significantly shorter (2.97 words on average).

Despite these differences — some are inevitable due to the unsupervised nature of pretraining datasets — the pretraining data is very helpful for our task as shown in the experimental results.

## 4 Method

To model segment-level caption generation, we adopt MASS-style pretraining (Song et al., 2019) with Transformer (Vaswani et al., 2017b) as the backbone architecture. For both pre-training and fine-tuning objectives, we have considered two variants: text-only and multi-modal. They are summarized in Table 2 and more details are given below.

### 4.1 Separate-Modality Architecture

Both ASR tokens and video segment features are given as input in the multimodal variants. We consider an architecture with a separate transformer for each modality (text or video), see Figure 2 for details. When available, the text and video encoders attend to each other at every layer using cross-modal attention, as in VILBERT (Lu et al., 2019). The text decoder attends over the final-layer output of both encoders. We discuss in more detail the differences between using a separate-modality architecture vs. a vanilla-Transformer approach for all modalities in Appendix A.2.

The inputs to the text encoder is the sum of three components: text token embeddings, positional embeddings and the corresponding style embeddings,<sup>5</sup> depending on the style of the text (ASR or Caption-like). The inputs to the video encoder

<sup>5</sup>This is similar to the way language-ID embeddings are used in machine translation.

could be either precomputed frame-level 2D CNN features or 3D CNN features, pretrained on the Kinetics (Carreira and Zisserman, 2017; Kay et al., 2017) data set. The visual features are projected with fully-connected layers to the same dimension as the text embeddings.

The main architecture we consider is a 2-layer encoder (E2), 6-layer decoder (D6) Transformer. We use **E2D6** to refer to the text-only version, and **E2vidD6** to refer to the multimodal version with an active video encoder. We also experiment with E2D2 and E2vidD2 (2-layer decoder).<sup>6</sup>

### 4.2 Pretraining with Text-only MASS

Text-only pretraining is essentially the unsupervised learning of the style transfer between ASR-style and caption-style texts using *unpaired* data sources: ASR strings from video segments in YT8M-cook or HowTo100M; and CAP-style instruction steps found in Recipe1M or HowTo100M. Just like using MASS for unsupervised machine translation involves pretraining the model on unpaired monolingual datasets, we alternate between ASR→ASR and CAP→CAP MASS steps during our pretraining stage, which does not require the “source” (ASR+Video) and “target” (CAP-style) data to be aligned.

In an ASR→ASR step, we mask a random subsequence of the ASR and feed the masked ASR to the text encoder. The text decoder must reconstruct the hidden subsequence while attending to the encoder output. A CAP→CAP step works similarly by trying to reconstruct a masked sequence of a CAP-style text. The encoder and decoder are trained jointly using teacher-forcing on the decoder. We denote this text-only strategy as **MASS** in the experiments.

### 4.3 Pretraining with Multimodal MASS

During multimodal pretraining, we alternate between text-only CAP→CAP MASS steps and multimodal MASS steps. During each multimodal MASS step ASR+video→ASR, we feed a masked ASR to the text-encoder and the co-occurring video features to the video-encoder. The text decoder must reconstruct the masked ASR subsequence. We denote this pretraining strategy as **MASSvid** in the experiments. This trains cross-modal attention between the text-encoder and video-encoder

<sup>6</sup>We found in a preliminary study that using 6-layer encoders did not improve performance for our application.

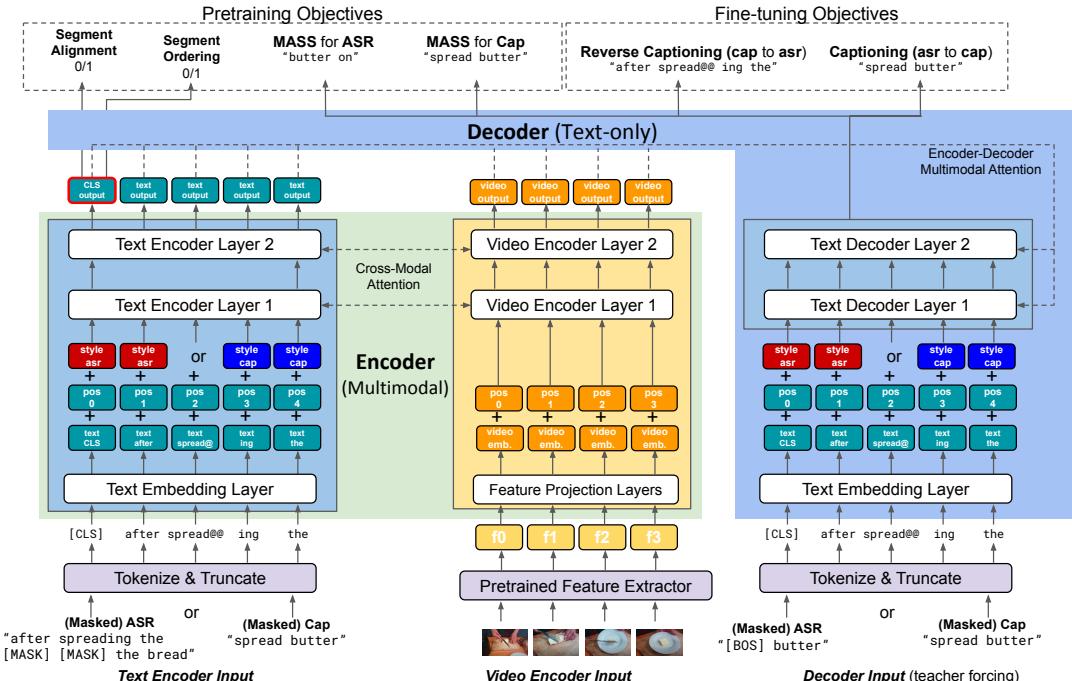


Figure 2: A diagram for the separate-modality architecture. It consists of a two-stream (text and video inputs) encoder with cross-modal attention and a text-only decoder, jointly trained using the MASS objective.

at every layer, jointly with the text decoder that attends to the output layer of both the text and video encoders.<sup>7</sup>

To force more cross-modal attention between encoder and decoder, we also investigate a strategy of hiding the text-encoder output from the decoder for some fraction of training examples. We refer to this strategy as **MASSdrop** in the experiments.

#### 4.4 Pretraining with Alignment and Ordering Tasks

We also explore encoder-only multimodal pretraining strategies. We take the last-layer representation for the **CLS** (beginning of sentence) token from the encoder, and add a multi-layer perceptron on top of it for binary predictions (Figure 2). Given a pair of ASR and video segment, we train the encoder to predict the following objectives:

- **Segment-Level Alignment.** An (ASR, video) pair is *aligned* if they occur in the same pre-training segment; negative examples are constructed by sampling pairs from the same video but at least 2 segments away.

- **Segment-Level Ordering.** We sample (ASR, video) pairs that are at least 2 segments away, and train the model to predict whether the ASR occurs before or after the video clip.

During this **MASSalign** pretraining stage, we alternate between two text-only MASS steps ( $\text{CAP} \rightarrow \text{CAP}$ ,  $\text{ASR} \rightarrow \text{ASR}$ ) and the two binary predictions (**Alignment** and **Ordering**) described above.

## 4.5 Finetuning on Video Captioning

For text-only finetuning, we feed ASR to the text encoder and the decoder has to predict the corresponding CAP ( $\text{ASR} \rightarrow \text{CAP}$ ). For multimodal finetuning, we also feed additional video representations to the video encoder ( $\text{ASR} + \text{video} \rightarrow \text{CAP}$ ). When finetuning a multimodal model from text-only pretraining, everything related to video (weights in the video encoder and any cross-modal attention modules) will be initialized randomly. In addition to these *uni-directional* (**UniD**) finetuning, we also experiment with several variants of *bidi-directional* (**BiD**) finetuning (Table 2). For instance, adding  $\text{CAP} \rightarrow \text{ASR}$  (predicting ASR from CAP) to text-only finetuning. In the experiments, we find some variants of bidirectional finetuning beneficial whether training from scratch or finetuning from a pretrained model.

<sup>7</sup>In preliminary experiments, we had attempted to directly adapt the MASS objective (Song et al., 2019) to video reconstruction — by masking a subsequence of the input video and making the video decoder reconstruct the input using the Noise Contrastive Estimator Loss (Sun et al., 2019a). Due to limited success, we did not further pursue this approach.

Pretraining Objectives			
Name	T	V	Input→Output
MASS	✓	✗	CAP→CAP, ASR→ASR
MASSvid	✓	✓	CAP→CAP, ASR+video→ASR
MASSdrop	✓	✓	CAP→CAP, ASR+video→ASR
MASSalign	✓	✓	CAP→CAP, ASR→ASR, ASR+video→{0, 1}
Finetuning Objectives			
Name	T	V	Input→Output
UniD	✓	✗	ASR→CAP
BiD	✓	✗	ASR→CAP, CAP→ASR
UniD	✓	✓	ASR+video→CAP
BiD	✓	✓	ASR+video→CAP, CAP→ASR
BiDalt	✓	✓	ASR+video→CAP, CAP+video→ASR

Table 2: Pretraining and Fine-tuning objectives. For each strategy, ✓ indicates whether the text (**T**) and video (**V**) encoders are active, followed by a summary of training objectives involved in one training step.

## 5 Experiments

### 5.1 Implementation Details

We tokenize ASR and CAP inputs using byte-pair-encoding subwords (Sennrich et al., 2015), and truncate them to 240 subwords. We truncate video sequences to 40 frames (40 seconds of video), compute the 128-dim features proposed by Wang et al. (2014) (which we will refer to as Compact 2D features), and project them to the embedding space using a two-layer perceptron with layer normalization and GeLU activations.

We instantiate the E2xDx models defined in Section 4.1 with 128-dimensional embeddings and 8 heads respectively for self-attention, encoder-decoder, and cross-modal attention modules. We define each epoch to be 3,125 iterations, where each iteration contains one repetition of each training step as represented in Table 2. We pretrain for 200 epochs and finetune for 30 epochs.

For evaluation, we consider BLEU-4 (Papineni et al., 2002), METEOR (Denkowski and Lavie, 2014), ROUGE-L (Lin and Och, 2004) and CIDEr (Vedantam et al., 2015) metrics.

Please refer to Appendix A.3 for full implementation details, hyperparameters and computation cost.

**Notes on ViTT evaluation:** With the exception of ROUGE-L, all other metrics are sensitive to short groundtruth. 67% of the groundtruth tags in ViTT have less than 4 words, where a perfect prediction will not yield a full score in, say, BLEU-4. Thus, we

focus mainly on ROUGE-L, report BLEU-1 instead of BLEU-4 for ViTT, and provide the other two metrics only as reference points.

We had originally decided to use videos with multiple annotations as validation and test data, so that we could explore evaluation with multiple reference groundtruth captions. But as annotators do not always yield the same set of segment boundaries, this became tricky. Instead, we simply treat each segment as a separate instance with one single reference caption. Note that all segments annotated for the same video will be in either validation or test to ensure no content overlap.

## 5.2 Main Results

We run several variants of our method on YouCook2, ViTT-All and ViTT-Cooking, using different architectures, modalities, pretraining datasets, pretraining and finetuning strategies.

### Comparing with other methods on YouCook2

For YouCook2, we report our method alongside several methods from the literature (Hessel et al., 2019; Sun et al., 2019b; Zhou et al., 2018c; Lei et al., 2020), as well as state-of-the-art concurrent work (Luo et al., 2020). The related work is provided for reference and to give a ballpark estimate of the relative performance of each method, but results are not always strictly and directly comparable. Beyond the usual sources of discrepancy in data processing, tokenization, or even different splits, an additional source of complication comes from the fact that videos are regularly deleted by content creators, causing video datasets to shrink over time. Additionally, when comparing to other work incorporating pretraining, we could differ in (videos available in) pretraining datasets, segmentation strategies, etc. To this end, we perform an extensive ablation study, which at least helps us to understand the effectiveness of different components in our approach.

**Effect of pretraining** The main experimental results for the three datasets we consider are summarized in Table 3 (YouCook2) and Table 4 (ViTT-All and ViTT-Cooking). Across all three datasets, the best performance is achieved by finetuning a multimodal captioning model under the *Multimodal Pretraining* condition. For instance, on YouCook2, E2vidD6-MASSvid-BiD improves over the no-pretraining model E2vidD6-BiD by 4.37 ROUGE-L, a larger improvement than UniViLM with pretraining (#5) vs without (#2) (Luo et al., 2020). This

Method	Input	Pretraining	BLEU-4	METEOR	ROUGE-L	CIDEr
Constant Pred (Hessel et al., 2019)	-	-	2.70	10.30	21.70	0.15
MART (Lei et al., 2020)	Video	-	8.00	15.90	-	0.36
EMT (Zhou et al., 2018c)	Video	-	4.38	11.55	27.44	0.38
CBT (Sun et al., 2019a)	Video	Kinetics + HowTo100M	5.12	12.97	30.44	0.64
AT (Hessel et al., 2019)	ASR	-	8.55	16.93	35.54	1.06
AT+Video (Hessel et al., 2019)	Video + ASR	-	9.01	17.77	36.65	1.12
UniViLM #1 (Luo et al., 2020)	Video	-	6.06	12.47	31.48	0.64
UniViLM #2 (Luo et al., 2020)	Video + ASR	-	8.67	15.38	35.02	1.00
UniViLM #5 (Luo et al., 2020)	Video + ASR	HowTo100M	10.42	16.93	38.02	1.20
<i>∅ Pretraining</i>						
E2D6-BiD	ASR	-	7.90	15.70	34.86	0.93
E2vidD6-BiD	Video + ASR	-	8.01	16.19	34.66	0.91
<i>Text Pretraining</i>						
E2D6-MASS-BiD	ASR	YT8M-cook + Recipe1M	10.60	17.42	38.08	1.20
E2vidD6-MASS-BiD	Video + ASR	YT8M-cook + Recipe1M	11.47	17.70	38.80	<b>1.25</b>
<i>Multimodal Pretraining</i>						
E2vidD6-MASSalign-BiD	Video + ASR	YT8M-cook + Recipe1M	11.53	17.62	<b>39.03</b>	1.22
E2vidD6-MASSvid-BiD	Video + ASR	YT8M-cook + Recipe1M	<b>12.04</b>	<b>18.32</b>	<b>39.03</b>	1.23
E2vidD6-MASSdrop-BiD	Video + ASR	YT8M-cook + Recipe1M	10.45	17.74	38.82	1.22
Human (Hessel et al., 2019)	Video + ASR	-	15.20	25.90	45.10	3.80

Table 3: Segment-level captioning results on YouCook2. We use YT8M-cook and Recipe1M for pretraining. The numbers for the related work (first group) are directly reported from the corresponding papers. The last line is an estimate of human performance as reported by Hessel et al. (2019), and can be taken as a rough upper bound of the best performance achievable.

improvement also holds in ViTT-Cooking (+4.22 in ROUGE-L) and ViTT-All (+2.97 in ROUGE-L). We do not observe consistent and significant trends among the different multimodal pretraining strategies: MASS pretraining with video (**MASSvid**), with video and droptext (**MASSdrop**), or with alignment tasks (**MASSalign**).<sup>8</sup> Furthermore, we observe that most of the pretraining improvement is achievable via text-only MASS pretraining. Across all three datasets, while *Multimodal Pretraining* (E2vidD6-MASSvid-BiD) is consistently better than *Text Pretraining* (E2vidD6-MASS-BiD), the differences are quite small (under one ROUGE-L point).

It’s worthy noting that for MASSalign, the best validation accuracies for the pretraining tasks are reasonably high: for YT8M-cook, we observed 90% accuracy for the alignment task, and 80% for the ordering task (for HowTo100M: 87% and 71.4%, respectively), where random guess would yield 50%. This suggests that our video features are reasonably strong, and the findings above are not due to weak visual representations.

<sup>8</sup>Limited improvement with MASSalign suggests that such alignment tasks are better suited for retrieval (Luo et al., 2020).

**Effect of other modeling choices** We experiment with 2-layer decoder (D2) vs 6-layer decoder (D6), combined with either unidirectional fine-tuning (**UniD**) or bidirectional fine-tuning (**BiD**). Table 5 shows ablation results of the four possible combinations when finetuning a multimodal model using text-only pretraining on YouCook2 (a more complete list of results can be found in Appendix A.5, showing similar trends). The D6xBiD combination tends to yield the best performance, with the differences among the four configurations being relatively small (under one ROUGE-L point). For visual features, we also explored using 3D features (Xie et al., 2018) instead of 2D features during finetuning (with no pretraining or text-only pre-training), and do not find much difference in model performance on YouCook2. As a result, we use the simpler 2D features in our multimodal pretraining. We leave more extensive experiments with visual features as future work.

**Generalization implications** An important motivation for constructing the ViTT dataset and evaluating our models on it has been related to generalization. Since the YouCook2 benchmark is restricted to a small number of cooking recipes, there is little to be understood about how well models

Method	Input	ViTT-All				ViTT-Cooking			
		BLEU-1	METEOR	ROUGE-L	CIDEr	BLEU-1	METEOR	ROUGE-L	CIDEr
Constant baseline (“intro”)	-	1.42	3.32	11.15	0.28	1.16	2.93	10.21	0.25
<i>∅ Pretraining</i>									
E2D6-BiD	ASR	19.60	9.12	27.88	0.68	20.77	10.08	28.63	0.72
E2vidD6-BiD	Video + ASR	19.49	9.23	28.53	0.69	20.45	9.88	28.88	0.69
<i>Text Pretraining</i>									
E2D6-MASS-BiD	ASR	21.93	10.60	30.45	0.79	24.79	12.25	32.40	0.88
E2vidD6-MASS-BiD	Video + ASR	22.44	10.83	31.27	0.81	24.22	12.22	32.60	0.89
<i>Multimodal Pretraining</i>									
E2vidD6-MASSalign-BiD	Video + ASR	22.31	10.66	31.13	0.79	<b>24.92</b>	12.25	33.09	<b>0.90</b>
E2vidD6-MASSvid-BiD	Video + ASR	<b>22.45</b>	10.76	<b>31.49</b>	0.80	24.87	<b>12.43</b>	32.97	<b>0.90</b>
E2vidD6-MASSdrop-BiD	Video + ASR	22.37	<b>11.00</b>	31.40	<b>0.82</b>	24.48	12.22	<b>33.10</b>	0.89
Human	Video + ASR	43.34	33.56	41.88	1.26	41.61	32.50	41.59	1.21

Table 4: Segment-level captioning results on ViTT. For ViTT-All we pretrain on HowTo100M and WikiHow; for ViTT-Cooking we pretrain on YT8M-cook and Recipe1M. We report baseline scores for predicting the most common caption “intro”. We also estimate the human performance as a rough upper bound (details in Supplementary Material A.1; Table 9).

Method	BLEU-4	METEOR	ROUGE-L	CIDEr
D2-UniD	10.84	17.39	38.24	1.16
D6-UniD	11.39	18.00	38.71	1.22
D2-BiD	11.38	<b>18.04</b>	38.67	1.19
D6-BiD	11.47	17.70	<b>38.80</b>	<b>1.25</b>
D6-BiDalt	11.07	17.68	38.43	1.22
D6-BiD (S3D)	<b>11.64</b>	18.04	38.75	1.24

Table 5: Ablation study on YouCook2. We finetune a multimodal captioning model (E2vid) with either 2-layer decoder (D2) or 6-layer decoder (D6) using YT8M-cook /Recipe1M for MASS pretraining, combined with either unidirectional (UniD) or bidirectional (BiD) finetuning. We find no significant difference between using 2D and 3D features (marked as S3D).

trained and evaluated on it generalize. In contrast, the ViTT benchmark has a much wider coverage (for both cooking-related videos and general instructional videos), and no imposed topic overlap between train/dev/test. As such, there are two findings here that are relevant with respect to generalization: (a) the absolute performance of the models on the ViTT benchmark is quite high (ROUGE-L scores above 0.30 are usually indicative of decent performance), and (b) the performance on ViTT vs. YouCook2 is clearly lower (31.5 ROUGE-L vs. 39.0 ROUGE-L, reflecting the increased difficulty of the new benchmark), but it is maximized under similar pretraining and finetuning conditions, which allows us to claim that the resulting models generalize well and are quite robust over a wide variety of instructional videos.

## 6 Conclusions

Motivated to improve information-seeking capabilities for videos, we have collected and annotated a new dense video captioning dataset, ViTT, which is larger with higher diversity compared to YouCook2. We investigated several multimodal pretraining strategies for segment-level video captioning, and conducted extensive ablation studies. We concluded that MASS-style pretraining is the most decisive factor in improving the performance on all the benchmarks used. Even more to the point, our results indicate that most of the performance can be attributed to leveraging the ASR signal. We achieve new state-of-the-art results on the YouCook2 benchmark, and establish strong performance baselines for the new ViTT benchmark, which can be used as starting points for driving more progress in this direction.

## Acknowledgements

We send warm thanks to Ashish Thapliyal for helping the first author debug his code and navigate the computing infrastructure, and to Sebastian Goodman for his technical help (and lightning fast responses!). We also thank the anonymous reviewers for their comments and suggestions.

## References

Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan.

2016. Youtube-8m: A large-scale video classification benchmark. *arXiv preprint arXiv:1609.08675*.
- Jean-Baptiste Alayrac, Piotr Bojanowski, Nishant Agrawal, Ivan Laptev Josef Sivic, and Simon Lacoste-Julien. 2016. Unsupervised learning from narrated instruction videos. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.
- João Carreira and Andrew Zisserman. 2017. Quo vadis, action recognition? a new model and the kinetics dataset. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4724–4733.
- Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2019. Uniter: Learning universal image-text representations. *arXiv preprint arXiv:1909.11740*.
- Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. 2018. Scaling egocentric vision: The EPIC-KITCHENS dataset. In *European Conference on Computer Vision (ECCV)*.
- Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the ninth workshop on statistical machine translation*, pages 376–380.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
- Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. *arXiv preprint arXiv:1905.03197*.
- Jack Hessel, Bo Pang, Zhenhai Zhu, and Radu Soricut. 2019. A case study on combining asr and visual features for generating instructional video captions. In *Proceedings of CoNLL*.
- Will Kay, João Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Apostol Natsev, Mustafa Suleyman, and Andrew Zisserman. 2017. The kinetics human action video dataset. *ArXiv*, abs/1705.06950.
- Juho Kim, Phu Tran Nguyen, Sarah Weir, Philip J Guo, Robert C Miller, and Krzysztof Z Gajos. 2014. Crowdsourcing step-by-step information extraction to enhance existing how-to videos. In *CHI*.
- Mahnaz Koupaee and William Yang Wang. 2018. Wikihow: A large scale text summarization dataset. *arXiv preprint arXiv:1810.09305*.
- Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. 2017. Dense-captioning events in videos. In *International Conference on Computer Vision (ICCV)*.
- Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. *ArXiv*, abs/1909.11942.
- Jie Lei, Liwei Wang, Yelong Shen, Dong Yu, Tamara L. Berg, and Mohit Bansal. 2020. Mart: Memory-augmented recurrent transformer for coherent video paragraph captioning. In *The 58th Annual Meeting of the Association for Computational Linguistics*.
- Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *ArXiv*, abs/1910.13461.
- Gen Li, Nan Duan, Yuejian Fang, Dixin Jiang, and Ming Zhou. 2019. Unicoder-vl: A universal encoder for vision and language by cross-modal pre-training. *arXiv preprint arXiv:1908.06066*.
- Chin-Yew Lin and Franz Josef Och. 2004. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 605. Association for Computational Linguistics.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *arXiv preprint arxiv:1907.11692*.
- Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems*, pages 13–23.
- Huaishao Luo, Lei Ji, Botian Shi, Haoyang Huang, Nan Duan, Tianrui Li, Xilin Chen, and Ming Zhou. 2020. Univilm: A unified video and language pre-training model for multimodal understanding and generation. *arXiv preprint arXiv:2002.06353*.
- Lauren E Margulieux, Mark Guzdial, and Richard Catrambone. 2012. Subgoal-labeled instructional material improves performance and transfer in learning to develop mobile applications. In *Conference on International Computing Education Research*.
- Javier Marin, Aritro Biswas, Ferda Ofli, Nicholas Hynes, Amaia Salvador, Yusuf Aytar, Ingmar Weber, and Antonio Torralba. 2019. Recipe1m+: A dataset for learning cross-modal embeddings for cooking recipes and food images. *IEEE transactions on pattern analysis and machine intelligence*.

- Antoine Miech, Jean-Baptiste Alayrac, Lucas Smaira, Ivan Laptev, and Andrew Zisserman Josef Sivic. 2020. End-to-end learning of visual representations from uncurated instructional videos. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.
- Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. 2019. Howto100m: Learning a text-video embedding by watching hundred million narrated video clips. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2630–2640.
- Celie O’Neil-Hart. 2018. Why you should lean into how-to content in 2018. [www.thinkwithgoogle.com/advertising-channels/video/self-directed-learning-youtube/](http://www.thinkwithgoogle.com/advertising-channels/video/self-directed-learning-youtube/). Accessed: 2019-09-03.
- Boxiao Pan, Haoye Cai, De-An Huang, Kuan-Hui Lee, Adrien Gaidon, Ehsan Adeli, and Juan Carlos Niebles. 2020. Spatio-temporal graph for video captioning with knowledge distillation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.
- Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. [https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language\\_understanding\\_paper.pdf](https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf).
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *ArXiv*, abs/1910.10683.
- Ramon Sanabria, Ozan Caglayan, Shruti Palaskar, Desmond Elliott, Loïc Barrault, Lucia Specia, and Florian Metze. 2018. How2: A large-scale dataset for multimodal language understanding. *arXiv preprint arXiv:1811.00347*.
- Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.
- Botian Shi, Lei Ji, Yaobo Liang, Nan Duan, Peng Chen, Zhendong Niu, and Ming Zhou. 2019. Dense procedure captioning in narrated instructional videos. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6382–6391.
- Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. Mass: Masked sequence to sequence pre-training for language generation. *arXiv preprint arXiv:1905.02450*.
- Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2019. Vi-bert: Pre-training of generic visual-linguistic representations. *arXiv preprint arXiv:1908.08530*.
- Chen Sun, Fabien Baradel, Kevin Murphy, and Cordelia Schmid. 2019a. Contrastive bidirectional transformer for temporal representation learning. *arXiv preprint arXiv:1906.05743*.
- Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. 2019b. Videobert: A joint model for video and language representation learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7464–7473.
- Hao Tan and Mohit Bansal. 2019. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017a. Attention is all you need. *ArXiv*, abs/1706.03762.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017b. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.
- Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575.
- Jiang Wang, Yang Song, Thomas Leung, Chuck Rosenberg, Jingbin Wang, James Philbin, Bo Chen, and Ying Wu. 2014. Learning fine-grained image similarity with deep ranking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1386–1393.
- Xin Wang, Jiawei Wu, Junkun Chen, Lei Li, Yuanfang Wang, and William Yang Wang. 2019. Vatex: A large-scale, high-quality multilingual dataset for video-and-language research. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4580–4590.
- Sarah Weir, Juho Kim, Krzysztof Z Gajos, and Robert C Miller. 2015. Learnersourcing subgoal labels for how-to videos. In *CSCW*.
- Saining Xie, Chen Sun, Jonathan Huang, Zhuowen Tu, and Kevin Murphy. 2018. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 305–321.

YouTubeBlog. 2017. You know what's cool? a billion hours. <https://youtube.googleblog.com/2017/02/you-know-whats-cool-billion-hours.html>. Accessed: 2020-06-23.

ZenithMedia. 2019. Online video viewing to reach 100 minutes a day in 2021. <https://www.zenithmedia.com/online-video-viewing-to-reach-100-minutes-a-day-in-2021/>. Accessed: 2020-06-23.

Ziqi Zhang, Yaya Shi, Chunfeng Yuan, Bing Li, Peijin Wang, Weiming Hu, and Zhengjun Zha. 2020. Object relational graph with teacher-recommended learning for video captioning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Qi Zheng, Chaoyue Wang, and Dacheng Tao. 2020. Syntax-aware action targeting for video captioning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Luowei Zhou, Chenliang Xu, and Jason J Corso. 2018a. Towards automatic learning of procedures from web instructional videos. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Luowei Zhou, Chenliang Xu, and Jason J. Corso. 2018b. YouCookII dataset. [http://youcook2.eecs.umich.edu/static/YouCookII/youcookii\\_readme.pdf](http://youcook2.eecs.umich.edu/static/YouCookII/youcookii_readme.pdf). Accessed: 2020-06-23.

Luowei Zhou, Yingbo Zhou, Jason J. Corso, Richard Socher, and Caiming Xiong. 2018c. End-to-end dense video captioning with masked transformer. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8739–8748.

Linchao Zhu and Yi Yang. 2020. Actbert: Learning global-local video-text representations. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

## A Appendix

Supplementary Material for “Multimodal Pretraining for Dense Video Captioning”.

### A.1 The ViTT dataset

**Sampling video for annotation.** The goal of the ViTT dataset design is to mirror topic distribution in the “wild”. Therefore, instead of starting from specific how-to instructions and searching for corresponding videos, we sampled videos from the validation set of the YouTube-8M dataset ([Abu-El-Haija et al., 2016](#)), a large-scale collection of YouTube videos with topical labels, subject to YouTube policies.

Exclusion criteria were lack of English ASR and the topic label “Game”. The latter was motivated by the fact that in this type of videos, the visual information predominantly features video games, while the ViTT dataset was intended to contain only videos with real-world human actions. Cooking videos can be easily identified by sampling videos that came with “Cooking” or “Recipe” topic labels. Given the convenience and the fact that much of prior work in this area had focused on cooking videos, approximately half of the dataset was designed to include cooking videos only, while the remaining videos would be randomly sampled non-cooking videos, as long as they were verified as instructional by human annotators.

**Annotation process** Annotators were presented with a video alongside its timestamped, automatic transcription shown in sentence-length paragraphs. They were asked to watch the video and first judge whether the video was instructional. For the purpose of our dataset, we determine that a video is instructional if it focuses on real-world human actions that are accompanied by procedural language explaining what is happening on screen, in reasonable details. Also for our purposes, instructional videos need to be grounded in real life, with a real person in the video exemplifying the action being verbally described.

For videos judged to be instructional, annotators were then asked to:

- Delimit the main segments of the video.
- Determine their start time if different from the automatically suggested start time (explained below).

- Provide a label summarizing or explaining the segment.

**Annotation guidelines** Annotators were instructed to identify video segments with two potential purposes:

- Allow viewers to jump straight to the start of a segment for rewatch.
- Present viewers with an index to decide whether to watch the video in full or directly skip to the segment of interest.

Our guidelines suggested a range of five to ten segments as long as the the structure and content of the video permitted. For short videos, the direction was to prioritize quality over quantity and to only define those segments that formed the narrative structure of the video, even if the resulting number of segments was below 5.

To help annotators determine segment start times, transcriptions were shown in “sentences”—we expected that sentence start times might be good candidates for segment start times. We obtained sentence boundaries automatically as follows. Given the stream of timestamped ASR tokens for a video, we first separated them into blocks by breaking two consecutive tokens whenever they were more than 2 seconds apart. We then used a punctuation prediction model to identify sentence boundaries in each resulting block. Each sentence was shown with the timestamp corresponding to its first token. Annotators were advised that transcriptions had been automatically divided into paragraphs that may or may not correspond to a video segment—if they decided that a segment started from a particular sentence, they could choose to use the start time of the sentence as the start time for the segment, or, if needed, they could put in an adjusted start time instead.

Once the start time had been identified, annotators were asked to provide a free-text label to summarize each segment. We instructed the annotators to use nouns or present participles (-ing form of verbs) to write the labels for the video segments, whenever possible. Additionally, we asked that the labels be succinct while descriptive, using as few words as possible to convey as much information as possible.

**Data statistics and post-processing** The resulting dataset consists of 8,169 instructional videos that received segment-level annotations, of which

3,381 are cooking-related. Overall there are an average of 7.1 segments per video (max: 19). Given our instructions, the descriptions are much shorter in lengths compared to a typical captioning dataset: on average there are 2.97 words per description (max: 16); 20% of the captions are single-word, 22% are two-words, and 25% are three words. We refer to these descriptions as “tags” given how short they are.

When possible, annotators were also asked to start and end the video with an opening and closing segment. As a result, most annotations start with an introduction segment: this accounts for roughly 11% of the 88455 segments in the dataset (“intro”: 8%, “introduction”: 2.3%). Note that while “intro” and “introduction” are clearly paraphrases of each other, an automatic metric will penalize a model predicting “intro” when the groundtruth is “introduction”. Similarly, the ending segment was described in several varieties: “outro”: 3.4%, “closing”: 1%, “closure”, “conclusion”, “ending”, “end of video”: each under 1%. Penalizing paraphrases of the ground truth is an inherent weakness of automatic metrics. To mitigate this, we decided to reduce the chance of this happening for the most frequent tags in the dataset. That is, in our experiments, we identified three groups of tags among the top-20 most frequent tags, and standardized them as follows.

intro	intro, introduction, opening
outro	outro, closing, closure, conclusion, ending, end of video, video closing
result	finished result, final result, results

Table 6: Standardization of top tags

Note that this does not mean we can solve this problem as a classification task like in visual question answering (VQA): overall, there are 56,027 unique tags with a vocabulary size of 12,509 for the 88,455 segments; 51,474 tags appeared only once in the dataset, making it infeasible to reduce the segment-level captioning problem into a pure classification task. Table 7 shows the top 10 most frequent tags after standardization.

**Estimate of human performance.** A subset of the candidate videos were given to three annotators<sup>9</sup>, to help us understand variations in human annotations. 5,840 videos received dense captioning

<sup>9</sup>A small set were unintentionally given to six annotators.

Tag	% of segments
intro	11.4
outro	6.6
result	0.9
ingredients	0.8
listing ingredients	0.2
supplies	0.2
mixing ingredients	0.2
materials	0.1
what you’ll need	0.1
lining the eyes	0.1

Table 7: 10 most frequent tags after standardization.

from exactly one annotator and were used as training data. Videos with more than one annotation were used as validation / test data. Note that not all the videos with multiple timeline annotations have exactly three sets of them — in fact, 1368 videos received 3-way segment-level annotations. This is because not all annotators agreed on whether a video was instructional. Computing annotator agreement for the annotated timelines is non-trivial. Here we focus on an estimate of tagging agreement when a pair of annotators agreed over the segment start time. Specifically, we go through each video that received multiple segment-level annotations. For each segment where two annotators chose the same ASR sentence as its starting point, we take the tags they produced for this segment and consider one of them as groundtruth, the other as prediction, and add that into our pool of (groundtruth, prediction) pairs. We can then compute standard automatic evaluations metrics over this pool. The results are as follows.

BLEU-1	METEOR	ROUGE-L	CIDEr
43.34	33.56	41.88	1.26

Table 8: Estimate of human performance for the segment-level captioning on ViTT-All (computed over 7528 pairs).

BLEU-1	METEOR	ROUGE-L	CIDEr
41.61	32.50	41.59	1.21

Table 9: Estimate of human performance for the segment-level captioning on ViTT-Cooking (computed over 2511 pairs).

Note that METEOR, and CIDEr scores are both penalized by the lack of n-grams for higher n. That

is, when both groundtruth and prediction are single-word, say, “intro”, this pair will not receive a full score from any of these metrics. But the ROUGE-L score is in the same ballpark as estimate of human performance in prior work (Hessel et al., 2019). One might note that perhaps this pool of label pairs contains a higher share of “intro”, since annotators might be more likely to agree over where an opening segment starts. Indeed, 20% of the time, one of the tags is “intro”. Interestingly, in spite of standardization of top tags, 14% of the time one tag is “intro”, the other tag is *not* “intro”: they can be less frequent paraphrases (e.g., “welcoming”, “greeting”, “opening and welcoming”) or something semantically different (e.g., “using dremel tool”).

## A.2 Separated vs. Concatenated-Modality Architecture

Prior work has explored both concatenating different modalities and feeding them into the same multimodal Transformer encoder (Sun et al., 2019b; Hessel et al., 2019), as well as separating them into unimodal transformers (Sun et al., 2019a; Lu et al., 2019). We opt for the separated architecture because it offers more flexibility. First, the concatenated architecture requires embedding the text and video features into the same space. When the video features are projected using a simple network, there is no guarantee that we can meaningfully project them into the text embedding space. VideoBERT (Sun et al., 2019b) gives more flexibility to the video embeddings by quantizing video features and learning an embedding for each codeword. However, the quantization step has subsequently been claimed to be detrimental (Sun et al., 2019a). Moreover, the concatenated architecture uses the same sets of forward and attention weights to process text and video, and performs layer normalization jointly between the two modalities, which is not necessarily meaningful. Finally, the separated architecture makes it easy to switch between variable length text-only, video-only, or text+video modalities, whereas concatenated architectures might rely on separating tokens, modalities embeddings, and using fixed sequence lengths (Luo et al., 2020).

## A.3 Additional Implementation Details

We optimize all models on a nVidia v100 GPU using the Adam optimizer with inverse square root schedule, batch size 32, warm-up period of 4,000

iterations, and maximum learning rate of 0.0001, following MASS (Song et al., 2019). The positional embeddings are initialized randomly. We use dropout and attention dropout with probabilities 0.1. With E2vidD6, pretraining takes 3-6 days depending on the objective and bidirectional finetuning takes up to 1.5 days, however those times could be improved by optimizing the data pipeline.

## A.4 Example Predictions

We show examples of **good** and **bad** predictions on YouCook2 (Figure 5 and ViTT-All (Figure 4 and 5). The captions are generated by E2vidD6-BiD (no pretraining) and E2vidD6-MASS-BiD (text-only MASS pretraining).

## A.5 Full result tables

We present here tables with all the ablation results that we run. There are two main takeaway messages from the results involving the pretraining approach: (a) the accuracy improvements, as measured across all the metrics we use, indicate the value of using a pretraining approach to this problem, specifically one that is capable of leveraging the ASR signals at both pretraining and finetuning stages, and (b) the training speedup achieved from pretraining is impressive, as a pretrained model converges much faster than training from scratch. This is especially visible on ViTT-All where finetuning after MASS pretraining reaches best ROUGE-L score at epoch 2, whereas it takes around 11 epochs to converge when training from scratch.

Sample Frame	Ground Truth	$\emptyset$ -Pretraining	MASS-Pretraining	Comments
	crush and grate the garlic	grate garlic and add to bowl ( <b>good</b> )	crush ginger and garlic( <b>good</b> )	ginger is correct despite not appearing in ground truth.
	crimp shut with fork	place the filling on the wrapper ( <b>ok</b> )	seal the edges of the wrapper ( <b>good</b> )	pretrained model is more specific
	place wings on the baking sheet and cook flipping	bake the pizza in the oven ( <b>bad</b> )	cook the wings on the grill ( <b>good</b> )	only pretrained model predicted correct food
	add the pork back into the hot oil	add the rice to the pot ( <b>bad</b> )	place the meat on the pan ( <b>good</b> )	$\emptyset$ model hallucinates the rice and pot
	add thyme bay leaves onion and clam juice and boil the mixture	add diced tomatoes tomato puree and mix well ( <b>bad</b> )	add thyme thyme onion and clam juice to the pot and stir ( <b>ok</b> )	$\emptyset$ model hallucinates a lot of nonexistent ingredients
	cook bacon in a pot with oil and pepper	add chopped tomatoes to pan and stir ( <b>bad</b> )	add bacon and stir ( <b>ok</b> )	both models missed oil and pepper (not mentioned in ASR)
	pour dressing on top of the salad and toss	add dressing to the bowl ( <b>good</b> )	serve the soup over the salad ( <b>bad</b> )	pretrained model referred to dressing as "soup"
	slice the ginger into pieces	slice a celery ( <b>bad</b> )	slice the chicken ( <b>bad</b> )	both models had wrong ingredients (ASR segment does not mention what is being sliced)

Figure 3: Example good and bad predictions on YouCook2. The pretrained model is generally but not always better. Note that there are no “intro” or “outro”-like labels on YouCook2 because the dataset was specifically curated to only contain actual recipe steps.

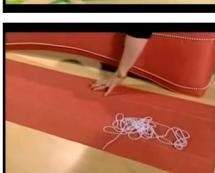
Sample Frame	Ground Truth	$\emptyset$ -Pretraining	MASS-Pretraining	Comments
	tightening extra loop	tightening the loop <b>(good)</b>	tightening the loop <b>(good)</b>	both models perform well
	adding eyeshadow	blending eye shadow <b>(good)</b>	applying eye shadow <b>(good)</b>	both models perform well
	showcasing the finished look	showing finished look <b>(good)</b>	finished showing look <b>(good)</b>	both models perform well
	rolling and folding the clay	rolling and blending <b>(ok)</b>	rolling and folding the clay <b>(good)</b>	MASS is a bit more specific
	highlighting brow bone	applying eye shadow <b>(ok)</b>	brushing on the brows <b>(good)</b>	MASS is a bit more specific
	covering the chicken and cooking	cooking the bread <b>(bad)</b>	cooking the chicken <b>(good)</b>	only MASS got the right ingredient
	connecting spray hose and sprayer	connecting the new cover <b>(ok)</b>	connecting the valve <b>(good)</b>	spray hose is more specific than valve
	implementing second layer	showing finished product <b>(ok)</b>	showing second layer <b>(good)</b>	MASS is more specific
	making decorative trim	cutting the edges <b>(good)</b>	cutting the fabric <b>(good)</b>	both models yield good predictions
	checking bleach container	outro <b>(bad)</b>	checking the container <b>(good)</b>	MASS is a bit more specific
	demonstrating the flip	checking the battery <b>(bad)</b>	flipping the board <b>(good)</b>	$\emptyset$ model got influenced by car mechanics tutorials
	tilting board	setting up the oven <b>(bad)</b>	turning the board <b>(good)</b>	$\emptyset$ overfitted on cooking videos

Figure 4: Example **good** predictions on ViTT-All (Part 1). The pretrained model is generally but not always better.

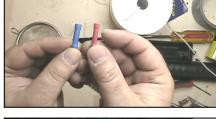
Sample Frame	Ground Truth	$\emptyset$ -Pretraining	MASS-Pretraining	Comments
	securing the bar in place	removing the cover <b>(bad)</b>	checking for the other side <b>(bad)</b>	predictions are not specific enough
	starting with unlocking bars	opening the box <b>(bad)</b>	pulling the car on <b>(bad)</b>	predictions are incorrect or not specific enough
	demonstrating technique	attaching paper <b>(bad)</b>	stamping paper <b>(good)</b>	the technique is about stamping the paper
	spritzing in additional water	pouring water into the water <b>(ok)</b>	adding water to water <b>(ok)</b>	understandable but ungrammatically
	checking for leaks	new new new new new new new new new new new new new new new new new new <b>(bad)</b>	checking the process <b>(ok)</b>	$\emptyset$ got into a loop, MASS not specific enough
	displaying materials needed	intro <b>(bad)</b>	removing paste <b>(ok)</b>	prediction makes sense because narrator is displaying thermal paste remover
	sketching on the swirls	drawing the lines <b>(good)</b>	drawing on the eyes <b>(bad)</b>	pretrained model overfitted on makeup tutorials
	crimping wire and completing project	attaching the screws <b>(bad)</b>	attaching the wire to the wire <b>(ok)</b>	both models have trouble with the concept of crimping a wire
	cutting with guide line	top of the top <b>(bad)</b>	explaining process <b>(ok)</b>	$\emptyset$ model got into a loop, MASS model is not specific enough

Figure 5: Example **ok** and **bad** predictions on ViTT (Part 2). The pretrained model is generally but not always better.

Method	Input	Pretraining	BLEU-4	METEOR	ROUGE-L	CIDEr
Constant Pred (Hessel et al., 2019)	-	-	2.70	10.30	21.70	0.15
MART (Lei et al., 2020)	Video	-	8.00	15.90	-	0.36
DPC (Shi et al., 2019)	Video + ASR	-	2.76	18.08	-	-
EMT (Zhou et al., 2018c)	Video	-	4.38	11.55	27.44	0.38
CBT (Sun et al., 2019a)	Video	Kinetics + HowTo100M	5.12	12.97	30.44	0.64
AT (Hessel et al., 2019)	ASR	-	8.55	16.93	35.54	1.06
AT+Video (Hessel et al., 2019)	Video + ASR	-	9.01	17.77	36.65	1.12
UniViLM #1 (Luo et al., 2020)	Video	-	6.06	12.47	31.48	0.64
UniViLM #2 (Luo et al., 2020)	Video + ASR	-	8.67	15.38	35.02	1.00
UniViLM #5 (Luo et al., 2020)	Video + ASR	HowTo100M	10.42	16.93	38.02	1.20
<i>∅ Pretraining</i>						
E2D2-UniD	ASR	-	7.42	15.15	33.26	0.85
E2D6-UniD	ASR	-	7.88	15.29	34.10	0.87
E2D2-BiD	ASR	-	6.85	15.64	34.26	0.91
E2D6-BiD	ASR	-	7.90	15.70	34.86	0.93
E2vidD2-UniD	Video + ASR	-	7.47	15.11	34.77	0.90
E2vidD6-UniD	Video + ASR	-	7.61	15.57	34.28	0.89
E2vidD2-BiD	Video + ASR	-	8.39	15.36	34.54	0.91
E2vidD6-BiD	Video + ASR	-	8.01	16.19	34.66	0.91
E2vidD2-BiDalt	Video + ASR	-	8.12	15.83	34.83	0.93
E2vid,D6-BiDalt	Video + ASR	-	7.70	16.11	34.78	0.91
E2vidD2-BiD (S3D)	Video + ASR	-	8.04	16.17	36.01	0.96
E2vidD6-BiD (S3D)	Video + ASR	-	7.91	16.28	35.23	0.93
<i>Text Pretraining</i>						
E2D2-MASS-UniD	ASR	YT8M-cook + Recipe1M	10.52	17.14	37.39	1.14
E2D6-MASS-UniD	ASR	YT8M-cook + Recipe1M	10.72	17.74	37.85	1.17
E2D2-MASS-BiD	ASR	YT8M-cook + Recipe1M	10.84	17.44	37.20	1.13
E2D6-MASS-BiD	ASR	YT8M-cook + Recipe1M	10.60	17.42	38.08	1.20
E2vidD2-MASS-UniD	Video + ASR	YT8M-cook + Recipe1M	10.84	17.39	38.24	1.16
E2vidD6-MASS-UniD	Video + ASR	YT8M-cook + Recipe1M	11.39	18.00	38.71	1.22
E2vidD2-MASS-BiD	Video + ASR	YT8M-cook + Recipe1M	11.38	18.04	38.67	1.19
E2vidD6-MASS-BiD	Video + ASR	YT8M-cook + Recipe1M	11.47	17.70	38.80	<b>1.25</b>
E2vid,D2-MASS-BiDalt	Video + ASR	YT8M-cook + Recipe1M	11.49	17.85	38.60	1.18
E2vid,D6-MASS-BiDalt	Video + ASR	YT8M-cook + Recipe1M	11.07	17.68	38.43	1.22
E2vidD2-MASS-BiD (S3D)	Video + ASR	YT8M-cook + Recipe1M	11.13	17.71	38.57	1.12
E2vidD6-MASS-BiD (S3D)	Video + ASR	YT8M-cook + Recipe1M	11.64	18.04	38.75	1.24
<i>Multimodal Pretraining</i>						
E2vidD2-MASSalign-BiD	Video + ASR	YT8M-cook + Recipe1M	11.54	17.57	37.70	1.15
E2vidD6-MASSalign-BiD	Video + ASR	YT8M-cook + Recipe1M	11.53	17.62	<b>39.03</b>	1.22
E2vidD2-MASSvid-BiD	Video + ASR	YT8M-cook + Recipe1M	11.17	17.71	38.32	1.17
E2vidD6-MASSvid-BiD	Video + ASR	YT8M-cook + Recipe1M	<b>12.04</b>	<b>18.32</b>	<b>39.03</b>	1.23
E2vidD2-MASSdrop-BiD	Video + ASR	YT8M-cook + Recipe1M	11.21	17.99	38.72	1.23
E2vidD6-MASSdrop-BiD	Video + ASR	YT8M-cook + Recipe1M	10.45	17.74	38.82	1.22
Human (Hessel et al., 2019)	Video + ASR	-	15.20	25.90	45.10	3.80

Table 10: Video Captioning Results on YouCook2. We use YT8M-cook/Recipe1M for pretraining. All video features are Compact 2D (Wang et al., 2014) except when marked as S3D (Xie et al., 2018).

Method	Input	Pretraining	BLEU-1	METEOR	ROUGE-L	CIDEr
Constant baseline (“intro”)	-	-	1.42	3.32	11.15	0.28
<i>Ø Pretraining</i>						
E2D2-UniD	ASR	-	17.94	8.55	27.06	0.64
E2D6-UniD	ASR	-	18.91	8.96	27.80	0.67
E2D2-BiD	ASR	-	18.81	8.82	27.63	0.65
E2D6-BiD	ASR	-	19.60	9.12	27.88	0.68
E2vidD2-UniD	Video + ASR	-	18.94	8.99	28.05	0.67
E2vidD6-UniD	Video + ASR	-	19.29	9.15	27.97	0.69
E2vidD2-BiD	Video + ASR	-	19.37	9.21	28.56	0.69
E2vidD6-BiD	Video + ASR	-	19.49	9.23	28.53	0.69
<i>Text Pretraining</i>						
E2D2-MASS-UniD	ASR	HowTo100M + WikiHow	21.53	10.24	29.95	0.77
E2D6-MASS-UniD	ASR	HowTo100M + WikiHow	22.09	10.58	30.67	0.79
E2D2-MASS-BiD	ASR	HowTo100M + WikiHow	20.73	10.20	30.15	0.76
E2D6-MASS-BiD	ASR	HowTo100M + WikiHow	21.93	10.60	30.45	0.79
E2vidD2-MASS-UniD	Video + ASR	HowTo100M + WikiHow	21.46	10.45	30.56	0.78
E2vidD6-UniD	Video + ASR	HowTo100M + WikiHow	22.21	10.75	30.86	0.81
E2vidD2-MASS-BiD	Video + ASR	HowTo100M + WikiHow	21.78	10.64	30.72	0.79
E2vidD6-MASS-BiD	Video + ASR	HowTo100M + WikiHow	22.44	10.83	31.27	0.81
<i>Multimodal Pretraining</i>						
E2vidD2-MASSalign-BiD	Video + ASR	HowTo100M + WikiHow	22.07	10.33	30.60	0.77
E2vidD6-MASSalign-BiD	Video + ASR	HowTo100M + WikiHow	22.31	10.66	31.13	0.79
E2vidD2-MASSvid-BiD	Video + ASR	HowTo100M + WikiHow	22.15	10.75	31.06	0.80
E2vidD6-MASSvid-BiD	Video + ASR	HowTo100M + WikiHow	<b>22.45</b>	10.76	<b>31.49</b>	0.80
E2vidD2-MASSdrop-BiD	Video + ASR	HowTo100M + WikiHow	21.84	10.55	31.10	0.79
E2vidD6-MASSdrop-BiD	Video + ASR	HowTo100M + WikiHow	22.37	<b>11.00</b>	31.40	<b>0.82</b>
Human estimate	Video + ASR	-	43.34	33.56	41.88	1.26

Table 11: Video captioning results on ViTT-All. We use HowTo100M/WikiHow for pretraining. We also estimate human performance (details in Appendix A.1; Table 9).

Method	Input	Pretraining	BLEU-1	METEOR	ROUGE-L	CIDEr
Constant baseline (“intro”)	-	-	1.16	2.93	10.21	0.25
<i>Ø Pretraining</i>						
E2D2-UniD	ASR	-	19.73	9.43	27.95	0.69
E2D6-UniD	ASR	-	20.24	9.93	28.59	0.71
E2D2-BiD	ASR	-	19.73	9.72	27.92	0.68
E2D6-BiD	ASR	-	20.77	10.08	28.63	0.72
E2vidD2-UniD	Video + ASR	-	19.97	9.75	28.30	0.69
E2vidD6-UniD	Video + ASR	-	20.46	9.93	28.62	0.69
E2vidD2-BiD	Video + ASR	-	20.60	10.08	29.45	0.71
E2vidD6-BiD	Video + ASR	-	20.45	9.88	28.88	0.69
<i>Text Pretraining</i>						
E2D2-MASS-UniD	ASR	YT8M-cook + Recipe1M	22.89	11.53	31.62	0.84
E2D6-MASS-UniD	ASR	YT8M-cook + Recipe1M	24.47	12.22	32.51	0.90
E2D2-MASS-BiD	ASR	YT8M-cook + Recipe1M	22.75	11.63	31.54	0.84
E2D6-MASS-BiD	ASR	YT8M-cook + Recipe1M	24.79	12.25	32.40	0.88
E2vidD2-MASS-UniD	Video + ASR	YT8M-cook + Recipe1M	23.86	11.85	32.32	0.86
E2vidD6-MASS-UniD	Video + ASR	YT8M-cook + Recipe1M	24.32	12.32	32.90	0.90
E2vidD2-MASS-BiD	Video + ASR	YT8M-cook + Recipe1M	22.93	11.68	32.15	0.87
E2vidD6-MASS-BiD	Video + ASR	YT8M-cook + Recipe1M	24.22	12.22	32.60	0.89
<i>Multimodal Pretraining</i>						
E2vidD2-MASSalign-BiD	Video + ASR	YT8M-cook + Recipe1M	24.02	11.91	32.73	0.86
E2vidD6-MASSalign-BiD	Video + ASR	YT8M-cook + Recipe1M	<b>24.92</b>	12.25	33.09	<b>0.90</b>
E2vidD2-MASSvid-BiD	Video + ASR	YT8M-cook + Recipe1M	24.15	12.10	32.96	0.88
E2vidD6-MASSvid-BiD	Video + ASR	YT8M-cook + Recipe1M	24.87	<b>12.43</b>	32.97	<b>0.90</b>
E2vidD2-MASSdrop-BiD	Video + ASR	YT8M-cook + Recipe1M	23.70	12.01	32.71	0.88
E2vidD6-MASSdrop-BiD	Video + ASR	YT8M-cook + Recipe1M	24.48	12.22	<b>33.10</b>	0.89
Human estimate	Video + ASR	-	41.61	32.50	41.59	1.21

Table 12: Video captioning results on ViTT-Cooking. We use YT8M-cook and Recipe1M for optional pretraining.

# Systematic Generalization on gSCAN with Language Conditioned Embedding

Tong Gao\* Qi Huang\* Raymond J. Mooney

Department of Computer Science

University of Texas at Austin

{gaotong, qhuang, mooney}@cs.utexas.edu

## Abstract

Systematic Generalization refers to a learning algorithm’s ability to extrapolate learned behavior to unseen situations that are distinct but semantically similar to its training data. As shown in recent work, state-of-the-art deep learning models fail dramatically even on tasks for which they are designed when the test set is systematically different from the training data. We hypothesize that explicitly modeling the relations between objects in their contexts while learning their representations will help achieve systematic generalization. Therefore, we propose a novel method that learns objects’ contextualized embedding with dynamic message passing conditioned on the input natural language and is end-to-end trainable with other downstream deep learning modules. To our knowledge, this model is the first one that significantly outperforms the provided baseline and reaches state-of-the-art performance on *grounded* SCAN (gSCAN), a grounded natural language navigation dataset designed to require systematic generalization in its test splits.

## 1 Introduction

Systematic Generalization refers to a learning algorithm’s ability to extrapolate learned behavior to unseen situations that are distinct but semantically similar to its training data. It has long been recognized as a key aspect of humans’ cognitive capacities (Fodor et al., 1988). Specifically, humans’ mastery of systematic generalization is prevalent in grounded natural language understanding. For example, humans can reason about the relations between all pairs of concepts from two domains, even if they have only seen a small subset of pairs during training. If a child observes “red squares”, “green squares” and “yellow circles”, he or she can

recognize “red circles” at their first encounter. Humans can also contextualize their reasoning about objects’ attributes. For example, a city being referred to as “the larger one” within a state might be referred to as “the smaller one” nationwide. In the past decade, deep neural networks have shown tremendous success on a collection of grounded natural language processing tasks, such as visual question answering (VQA), image captioning, and vision-and-language navigation (Hudson and Manning, 2018; Anderson et al., 2018a,b). Despite all the success, recent literature shows that current deep learning approaches are exploiting statistical patterns discovered in the datasets to achieve high performance, an approach that does not achieve systematic generalization. Gururangan et al. (2018) discovered that annotation artifacts like negation words or purpose clauses in natural language inference data can be used by simple text classification categorization model to solve the given task. Jia and Liang (2017) demonstrated that adversarial examples can fool reading comprehension systems. Indeed, deep learning models often fail to achieve systematic generalizations even on tasks on which they are claimed to perform well. As shown by Bahdanau et al. (2018), state-of-the-art Visual Questioning Answering (VQA) (Hudson and Manning, 2018; Perez et al., 2018) models fail dramatically even on a synthetic VQA dataset designed with systematic difference between training and test sets.

In this work, we focus on approaching systematic generalization in grounded natural language understanding tasks. We experiment with a recently introduced synthetic dataset, *grounded* SCAN (gSCAN), that requires systematic generalization to solve (Ruis et al., 2020). For example, after observing how to “walk hesitantly” to a target object in a grid world, the learning agent is tested with instruction that requires it to “pull hesitantly”, therefore testing its ability to generalize adverbs to

(\* ) denotes co-first authorship, authors contribute equally and are listed in alphabetical order.

unseen adverb-verb combinations.

When presented with a world of objects with different attributes, and natural language sentences that describe such objects, the goal of the model is to generalize its ability to understand unseen sentences describing novel combinations of observed objects, or even novel objects with observed attributes. One of the essential steps in achieving this goal is to obtain good object embeddings to which natural language can be grounded. By considering each object as a bag of its descriptive attributes, this problem is further transformed into learning good representations for those attributes based on the training data. This requires: 1) learning good representations of attributes whose actual meanings are contextualized, for example, "smaller" and "lighter", etc.; 2) learning good representations for attributes so that conceptually similar attributes, e.g., "yellow" and "red", have similar representations. We hypothesize that explicitly modeling the relations between objects in their contexts, i.e., learning contextualized object embeddings, will help achieve systematic generalization. This is intuitively helpful for learning concepts with contextualized meaning, just as learning to recognize the "smaller" object in a novel pair requires experience of comparison between semantically similar object pairs. Learning contextualized object embeddings can also be helpful for obtaining good representations for semantically similar concepts when such concepts are the only difference between two contexts. Inspired by [Hu et al. \(2019\)](#), we propose a novel method that learns an object's contextualized embedding with dynamic message passing conditioned on the input natural language. At each round of message passing, our model collects relational information between each object pair, and constructs an object's contextualized embedding as a weighted combination of them. Such weights are dynamically computed conditioned on the input natural sentence. This contextualized object embedding scheme is trained end-to-end with downstream deep modules for specific grounded natural language processing tasks, such as navigation. Experiments show that our approach significantly outperforms a strong baseline on gSCAN.

## 2 Related Work

Research on deep learning models' systematic generalization behavior has gained traction in recent years, with particular focus on natural language

processing tasks.

### 2.1 Compositionality

An idea that is closely related to systematic generalization is compositionality. [Kamp and Partee \(1995\)](#) define the principle of compositionality as "The meaning of a whole is a function of the meanings of the parts and of the way they are syntactically combined". [Hupkes et al. \(2020\)](#) synthesizes different interpretations of this abstract principle into 5 theoretically grounded tests to evaluate a model's ability to represent compositionality: 1) Systematicity: if the model can systematically recombine known parts and rules; 2) Productivity: if the model can extend their predictions beyond what they have seen in the training data; 3) Substitutivity; if the model is robust to synonym substitutions; 4) Localism: if the model's composition operations are local or global; and 5) Overgeneralisation: if the model favors rules or exceptions during training. The gSCAN dataset focuses more on capturing the first three tests in a grounded natural language understanding setting, and our proposed model achieves significant performance improvement on test sets relating to systematicity and substitutivity.

### 2.2 Systematic Generalization Datasets

Many systematic generalization datasets have been proposed in recent years ([Bahdanau et al., 2018](#); [Chevalier-Boisvert et al., 2019](#); [Hill et al., 2020](#); [Lake and Baroni, 2017](#); [Ruis et al., 2020](#)). This paper is conceptually most related to the SQOOP dataset proposed by [Bahdanau et al. \(2018\)](#), the SCAN dataset proposed by [Lake and Baroni \(2017\)](#), and the gSCAN dataset proposed by [Ruis et al. \(2020\)](#).

The SQOOP dataset consists of a random number of MNIST-style alphanumeric characters scattered in an image with specific spatial relations ("left", "right", "up", "down") among them ([Bahdanau et al., 2018](#)). The algorithm is tested with a binary decision task of reasoning about whether a specific relation holds between a pair of alphanumeric characters. Systematic difference is created between the testing and training set by only providing supervision on relations for a subset of digit pairs to the learner, while testing its ability to reason about relations between unseen alphanumeric character pairs. For example, the algorithm is tested with questions like "is S above T" while it never sees a relation involving both S and T during train-

ing. Therefore, to fully solve this dataset, it must learn to generalize its understanding of the relation “above” to unseen pairs of characters. [Lake and Baroni \(2017\)](#) proposed the SCAN dataset and its related benchmark that tests a learning algorithm’s ability to perform compositional learning and zero-shot generalization on a natural language command translation task . Given a natural language command with a limited vocabulary, an algorithm needs to translate it into a corresponding action sequence consisting of action tokens from a finite token set. Compared to SQuOOP, SCAN tests the algorithm’s ability to learn more complicated linguistic generalizations like “walk around left” to “walk around right”. SCAN also ensures that the target action sequence is unique, and an oracle solution exists by providing an interpreter function that can unambiguously translate any given command to its target action sequence.

Going beyond SCAN that focuses purely on syntactic aspects of systematic generalization, the gSCAN dataset proposed by [Ruis et al. \(2020\)](#) is an extension of SCAN. It contains a series of systematic generalization tasks that require the learning agent to ground its understanding of natural language commands in a given grid world to produce the correct action token sequence. We choose gSCAN as our benchmark dataset, as its input command sentences are linguistically more complex, and requires processing multi-modal input.

### 2.3 Systematic Generliazation Algorithms

[Bahdanau et al. \(2018\)](#) demonstrated that modular networks, with a carefully chosen module layout, can achieve nearly perfect systematic generalization on SQuOOP dataset. Our approach can be considered as a conceptual generalization of theirs. Each object’s initial embedding can be considered as a simple affine encoder module, and we learn the connection scheme among these modules conditioned on natural language instead of hand-designing it. [Gordon et al. \(2019\)](#) proposed solving the SCAN benchmark by hard-coding their model to be equivariant to all permutations of SCAN’s verb primitives. [Andreas \(2020\)](#) proposed GECA (“Good-Enough Compositional Augmentation”) that systematically augments the SCAN dataset by identifying sentence fragments with similar syntactic context, and permuting them to generate novel training examples. This line of permutation-invariant approaches is shown to not generalize

well on the gSCAN dataset ([Ruis et al., 2020](#)). At the time of submission, our method was the first to outperform the strong baseline provided in the gSCAN benchmark, and also the first one to apply language-conditioned message passing to learn contextualized input embedding for systematic generalization tasks. Concurrent to our work, [Kuo et al. \(2020\)](#) proposed a family of parse-tree-based compositional RNN networks to enable systematic generalization, and heavily relies on off-the-shelf parsers to produce the network hierarchy. [Heinze-Deml and Bouchacourt \(2020\)](#) use an attention-based prediction of the target object’s location as an auxiliary training task to regularize the model. However, it only improves over the baseline model in [Ruis et al. \(2020\)](#) in a limited subset of test splits. For completeness, we also compare our model’s result with the above two concurrent works.

## 3 Problem Definition & Algorithm

### 3.1 Task Definition

gSCAN contains a series of systematic generalization tasks in a grounded natural language understanding setting. In gSCAN, the learning agent is tested with the task of following a given natural language instruction to navigate in a two-dimensional grid world with objects. This is achieved in the form of generating a sequence of action tokens from a finite action token set  $\mathcal{A} = \{\text{walk}, \text{push}, \text{pull}, \text{stay}, L_{\text{turn}}, R_{\text{turn}}\}$  that brings the agent from its starting location to the target location. An object in gSCAN’s world state is encoded with a one-hot encoding describing its attributes in three property types: 1) color  $\mathcal{C} = \{\text{red}, \text{green}, \text{blue}, \text{yellow}\}$  2) shape  $\mathcal{S} = \{\text{circle}, \text{square}, \text{cylinder}\}$  3) size  $\mathcal{D} = \{1, 2, 3, 4\}$ . The agent is also encoded as an “object” in the grid world, with properties including orientation  $\mathcal{O} = \{\text{left}, \text{right}, \text{up}, \text{down}\}$  and a binary variable  $\mathcal{B} = \{\text{yes}, \text{no}\}$  denoting the presence of the agent. Therefore, the whole grid is represented as a tensor  $x^S \in \mathcal{R}^{d \times d \times c}$ , where  $d$  is the dimension of the grid, and  $c = |\mathcal{C}| + |\mathcal{S}| + |\mathcal{D}| + |\mathcal{O}| + |\mathcal{B}|$ . Mathematically, given an input tuple  $x = (x^c, x^S)$ , where  $x^c = \{x_1^c, x_2^c, \dots, x_n^c\}$  represents the navigation instruction, the agent needs to predict the correct output action token sequence  $y = \{y_1, y_2, \dots, y_m\}$ . Despite its simple form, this task is quite challenging. For one, generating the correct action token sequence requires understanding the instruction within the context of the agent’s

current grid world. It also involves connecting specific instructions with complex dynamic patterns. For example, “pulling” a square will be mapped to a “pull” command when the square has a size of 1 or 2, but to “pull pull” when the square has a size of 3 or 4 (a “heavy” square); “move cautiously” requires the agent to turn left then turn right before making the actual move. gSCAN also introduces a series of test sets that have systematic differences from the training set. Computing the correct action token sequences on these test sets requires the model to learn to combine seen concepts into novel combinations, including novel object property combinations, novel contextual references, etc..

### 3.2 Model Architecture

The overview of our model architecture is shown in Figure 1. At the highest level, it follows the same encoder-decoder framework used by the baseline model in Ruis et al. (2020) to extract information from the input sentence/grid-world representation and to output navigation instructions. However, there is a paradigm shift in how we represent and encode the grid world. Instead of viewing the grid world as a whole, we treat it as a collection of objects whose semantic meanings should be contextualized by their relations with one another. We also hypothesize that inter-object relations that are salient in a given grid world can be inferred from the accompanied natural language instruction. Therefore, we expand the vanilla CNN-based grid world encoder with a message passing module guided by the accompanied natural language instruction to obtain the contextualized grid-world embedding.

#### 3.2.1 Input Extraction

Given the input sentence and the grid world state, we first project them into higher dimensional embedding. For the input instruction  $I = \{w_1, w_2, \dots, w_S\}$  where  $w_i$  is the embedding vector of word  $i$ , following the practice of Ruis et al. (2020) and Hu et al. (2019), we first encode it as the hidden states  $\{h_s\}_{s=1}^S$  and the summary vector  $s$  obtained by feeding the input  $I$  to a Bi-LSTM as:

$$[h_1, h_2, \dots, h_S] = BiLSTM(I) \text{ and } s = [h_1; h_S] \quad (1)$$

Where we use semi-colon to represent concatenation, and  $h_i = [\overrightarrow{h}_i; \overleftarrow{h}_i]$  is the concatenation of the forward and backward direction of the LSTM hidden state for input word  $i$ . For each round of mes-

sage passing between the objects embedding, we further apply a transformation using a multi-step textual attention module similar to that of Hudson and Manning (2018) and Hu et al. (2018) to extract the round-specific textual context. Given a round-specific projection matrix  $W_2^t$ , the textual attention score for word  $i$  at message passing round  $t$  is computed as:

$$\alpha_{t,i} = \underset{s}{softmax}(W_1(h_i \odot (W_2^t \text{ReLU}(W_3 s)))) \quad (2)$$

The final textual context embedding for message passing round  $t$  is computed as:

$$c_t = \sum_{i=1}^S \alpha_{t,i} \cdot h_i \quad (3)$$

Details of the message passing mechanism will be described in later sections.

As for the grid-world representation, from each grid, we extract one-hot representations of color  $\mathcal{C}$ , shape  $\mathcal{S}$ , size  $\mathcal{D}$  and agent orientation  $\mathcal{O}$ , and embed each property with a 16-dimensional vector. We finally concatenate them back into one vector and use this vector as the object’s local embedding.

#### 3.2.2 Language-conditioned Message Passing

After extracting a textual context embedding and the objects’ local embedding, we perform a language-conditioned iterative message passing for  $T$  rounds to obtain the contextualized object embeddings, where  $T$  is a hyper-parameter.

1) Denoting the extracted object local embedding as  $x_i^{loc}$ , and previous round’s object context embedding as  $x_{i,t-1}^{ctx}$ , we first construct a fused representation of an object  $i$  at round  $t$  by concatenating its local, context embedding as well their element-wise product:

$$x_{i,t}^{fuse} = [x_i^{loc}, x_{i,t-1}^{ctx}, (W_4 x_i^{loc}) \odot (W_5 x_{i,t-1}^{ctx})] \quad (4)$$

We use an object’s local embedding to initialize its context embedding at round 0.

2) For each pair of objects  $(i, j)$ , we use their fused representations, together with this round’s textual context embedding to compute their message passing weight as:

$$w_{i,j}^t = \underset{s}{softmax}(W_6 x_{j,t}^{fuse})^T ((W_7 x_{i,t}^{fuse}) \odot (W_8 c_t)) \quad (5)$$

Note that the computation of the raw weight logits is asymmetric.

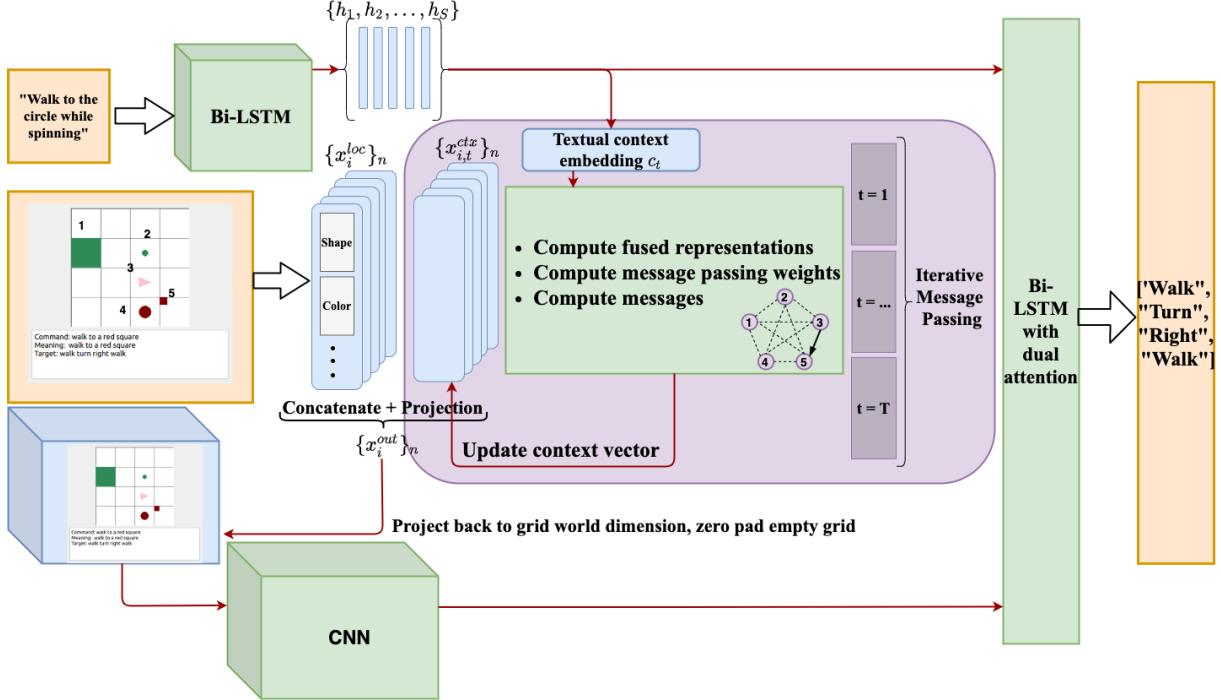


Figure 1: Model Overview

3) We consider all the objects in a grid world as nodes, and they together form a complete graph. Each node  $i$  computes its message to receiver node  $j$  as:

$$m_{i,j}^t = w_{i,j}^t \cdot ((W_9 x_{i,t}^{fuse} \odot (W_{10} c_t)) \quad (6)$$

and each receiver node  $j$  updates its context embedding as:

$$x_{j,t}^{ctx} = W_{11}[x_{j,t-1}^{ctx}; \sum_{i=1}^N m_{i,j}^t] \quad (7)$$

After  $T$  rounds of iterative message passing, the final contextualized embedding for object  $i$  will be:

$$x_i^{out} = W_{12}[x_i^{loc}; x_{i,T}^{ctx}] \quad (8)$$

### 3.2.3 Encoding the Grid World

After obtaining contextualized embeddings for all objects in a grid world  $x^s$  as  $\{x^{out}\}_n = \{x_1^{out}, x_2^{out}, \dots, x_n^{out}\}$  each of dimensionality  $\mathcal{R}^{out}$ , we map them back to their locations in the grid world, and construct a new grid world representation  $X^{s'} \in \mathcal{R}^{d \times d \times out}$  by zero-padding cells without any object. This is then fed into three parallel single convolutional layers with different kernel sizes to obtain a grid world's embedding at multiple scales, as done by Wang and Lake (2019). The final grid world encoding is as follows:

$$H^s = [H_1^s; H_2^s; H_3^s], \quad H_i^s = Conv_i(X^{s'}) \quad (9)$$

where  $Conv_i$  denotes the  $i$ th convolutional network, and  $H^s \in \mathcal{R}^{d^2 \times hid}$ .

### 3.2.4 Decoding Action Sequences

We use a Bi-LSTM with multi-modal attention to both the grid world embedding and the input instruction embedding to decode the final action sequence, following the baseline model provided by Ruis et al. (2020). At each step  $i$ , the hidden state of the decoder  $h_i^d$  is computed as:

$$h_i^d = LSTM([e_i^d; c_i^c; c_i^s], h_{i-1}^d) \quad (10)$$

where  $e_i^d$  is the embedding of the previous output action token  $y_{i-1}$ ,  $c_i^c$  is the instruction context computed with attention over textual encoder's hidden states  $[h_1^c, h_2^c, \dots, h_S^c]$ , and  $c_i^s$  is the grid world context computed with attention over all locations in the grid world embedding  $H^S$ . We set the decoder's hidden size to 64 so that it aligns with the textual encoder, and use the attention implementation proposed by Bahdanau et al. (2016). The instruction context is computed as:

$$e_{ij}^c = v_c^T \tanh W_c(h_{i-1}^d + h_j^c) \quad (11)$$

$$\alpha_{ij}^c = \frac{\exp(e_{ij}^c)}{\sum_{j=1}^S \exp(e_{ij}^c)} \quad (12)$$

$$c_i^c = \sum_{j=1}^S \alpha_{ij}^c h_j^c, \forall j \in \{1, 2, \dots, S\} \quad (13)$$

Similarly, the grid world context is computed as:

$$e_{ij}^s = v_s^T \tanh W_s(h_{i-1}^d + c_i^c) \quad (14)$$

$$\alpha_{ij}^s = \frac{\exp(e_{ij}^s)}{\sum_{j=1}^{d^2} \exp(e_{ij}^s)} \quad (15)$$

$$c_i^s = \sum_{j=1}^{d^2} \alpha_{ij}^s h_j^s, \forall j \in \{1, 2, \dots, d^2\} \quad (16)$$

where  $v_s, v_c, W_c, W_s$  are learnable parameters, and  $h_j^s$  is the embedding of grid  $j$  obtained from  $H^S$ . The distribution of next action token can then be computed as  $p(y_i|x, y_1, y_2, \dots, y_{i-1}) = \text{softmax}(W_o h_i^d)$ .

## 4 Experimental Evaluation

### 4.1 Methodology & Implementation

We run experiments to test the hypothesis that contextualized embeddings help systematic generalization<sup>1</sup>. Since this task has a limited vocabulary size, word-level accuracy is no longer a proper metric to reflect the model’s performance. We follow the baseline and use the exact match percentage as our metric, where an exact match means that the produced action token sequence is exactly the same as the target sequence. We compare our model with the baseline on different test sets, and use early stopping based on the exact match score on the validation set. We set the learning rate as 1e-4, decaying by 0.9 every 20,000 steps. We choose the number of message passing iterations to be 4. Our model is trained for 6 separate runs, and the average performance as well as the standard deviation are reported. Our encoder/decoder model is implemented in PyTorch (Paszke et al., 2017) and the message passing graph network is backed by DGL (Wang et al., 2020). For comparison, we use test set, validation set, and baseline model released by Ruis et al. (2020).

### 4.2 Results

Table 1 is an overview of 7 test splits used for evaluation, and table 2 shows our experiment results as well as other models’ performance for comparison. In the following sections, we present the results on each systematic generalization test split, and also introduce the configuration of test splits. Note that test **split A** is a random split set that has no systematic difference from the training set.

<sup>1</sup>Code is available [here](#)

**Split B:** This tests the model’s ability to generalize to navigation in a novel direction. For example, a testing example would require the agent to move to a target object that is to its south-west, even though during training target objects are never placed south-west of the agent. Although our model manages to predict some correct action sequences compared to the baseline’s complete failure, our model still fails on the majority of cases. We further analyze the failure on Split B in the discussion section.

**Split C, G:** Split C tests the model’s ability to generalize to novel contextual references. In the training set, a circle of size 2 is never referred to as “the small circle”, while in the test set the agent needs to generalize the notion “small” to it based on its size comparison with other circles in the grid world. The message passing mechanism helps the model comprehend the relative sizes of objects, and boost the performance on split C. Besides, our model shows promising results on exploring the interrelationship between an agent and other objects in the scene, as well as learning abstract concepts by contextual comparison as shown in split G. This test split asks the model to push a square of size 3. An object with the size of 3 or 4 is defined as “heavy”, according to the configuration, and requires two consecutive push/pull actions applied on it before it actually moves. The challenge here is that the model has been trained to “pull” heavy squares and “push” squares with size of 4, but was never trained to “push” a size-3 square. Thus, it needs to generalize the concept of “heavy” and act accordingly.

**Split D, E:** Split D and E are similar, as they both define the target object with novel combinations of color and shape. Split E is generally easier because the target object, a yellow square, appears as the target in training examples, but is only referred to as “the square”, “the smaller square”, or “the bigger square”. Split D increases the difficulty by referring to the red square, which never appears in the training set as a target but does appear as a background object. We find that while the baseline model understands the concept of “square”, it gets confused by target objects with a new color-shape combination. In contrast, our model can generalize to novel compositions of object properties and correctly find the target object, performing significantly better on these two splits.

**Split F:** This split is designed to test the model’s

Split	Description
A: Random	Randomly split test sets
B: Novel Direction	Target object is to the South-West of the agent
C: Relativity	Target object is a size 2 circle, referred to with the small modifier
D: Red Squares	Red squares are the target object
E: Yellow Squares	Yellow squares are referred to with a color and a shape at least
F: Adverb to Verb	All examples with the adverb ‘while spinning’ and the verb ‘pull’
G: Class Inference	All examples where the agent needs to push a square of size 3

Table 1: Description of test splits

Split	Baseline	Kuo et al. (2020)	Heinze-Deml and Bouchacourt (2020)	Ours
A: Random	$97.69 \pm 0.22$	97.32	$94.19 \pm 0.71$	<b><math>98.6 \pm 0.95</math></b>
B: Novel Direction	$0 \pm 0$	<b>5.73</b>	N/A	$0.16 \pm 0.12$
C: Relativity	$35.02 \pm 2.35$	75.19	$43.43 \pm 7.0$	<b><math>87.32 \pm 27.38</math></b>
D: Red Squares	$23.51 \pm 21.82$	80.16	<b><math>81.07 \pm 10.12</math></b>	$80.31 \pm 24.51$
E: Yellow Squares	$54.96 \pm 39.39$	95.35	$86.45 \pm 6.28$	<b><math>99.08 \pm 0.69</math></b>
F: Adverb to Verb	$22.7 \pm 4.59$	0	N/A	<b><math>33.6 \pm 20.81</math></b>
G: Class Inference	$92.95 \pm 6.75$	98.63	N/A	<b><math>99.33 \pm 0.46</math></b>

Table 2: Exact match accuracy of test splits

ability to generalize to novel adverb-verb combinations, where the model is tested under different situations but always with the terms “while spinning” and “pull” in the commands. However, they never appear in the training set together, consequently the model needs to generalize to this novel combination of adverb and verb. The results shows that our model does a bit better than the baseline, but suffers from high variance across different runs.

Comparing to the two concurrent works Kuo et al. (2020) and Heinze-Deml and Bouchacourt (2020), our model yields better performance in general. Notice that Heinze-Deml and Bouchacourt (2020) and our model also report the standard deviation of multiple runs, while Kuo et al. (2020) does not.

### 4.3 Discussion

**Model Comparison.** We reveal the strength of our model by analyzing two test examples where it succeeds and the baseline fails. For each example, we visualize the grid world that the agent is in, where each cell is colored with different grey-scale levels indicating its assigned attention score. For reader’s convenience, we also visualize the model’s prediction and the target sequence by the red path and green path, respectively.

Figure 2 from split G visualizes the prediction sequence as well as the attention weights generated by the baseline. The baseline attends to the position of the target object but is unable to capture the dynamic relationship between the target object and the green cylinder. It tries to push the target object over it, while our model correctly predicts

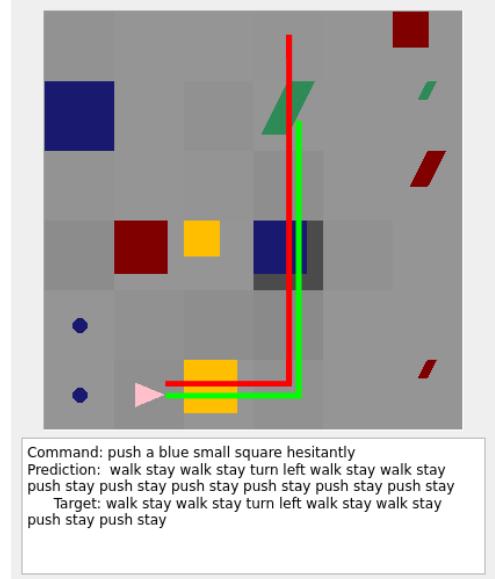


Figure 2: While the target is correctly chosen, the baseline did not stop pushing even after encountering an obstacle.

the incoming collision and stops at the right time.

Another example from split D where our model outperforms the baseline is shown in Figure 3. The baseline model incorrectly attends to two small blue squares and picks one as the target rather than the correct small red square. Note that the model has seen blue and green squares as targets in the training set, but has never seen a red square. This is a common mistake since the baseline struggles to choose target objects with novel property combinations when there are similar objects in the scene that were seen during training. On the contrary,

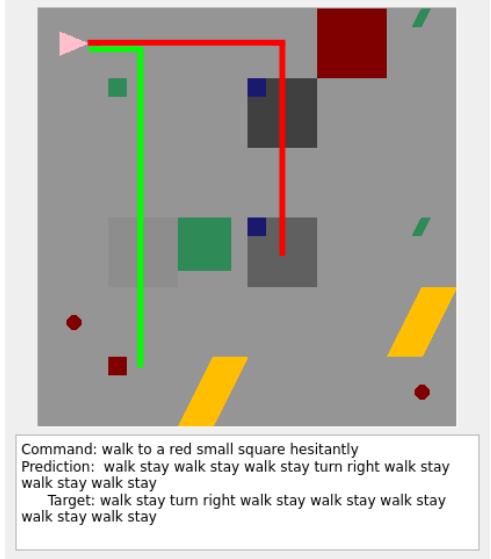


Figure 3: Baseline cannot distinguish the correct square from similar candidates.

our model handles these cases well, demonstrating its ability to generalize to novel color-shape combinations with the help of contextualized object embedding.

Split	No Message Passing	Full Model
A	$91.07 \pm 0.61$	$98.6 \pm 0.95$
B	$0.16 \pm 0.04$	$0.16 \pm 0.12$
C	$50.26 \pm 5.9$	$87.32 \pm 27.38$
D	$35.95 \pm 13.13$	$80.31 \pm 24.51$
E	$44.18 \pm 24.56$	$99.08 \pm 0.69$
F	$44.82 \pm 1.95$	$33.6 \pm 20.81$
G	$93.02 \pm 0.33$	$99.33 \pm 0.46$

Table 3: Ablation study

**Ablation Study.** We conduct an ablation study to test the significance of the language-conditioned message passing component in our network. We built a model whose architecture and hyper-parameters are the same as our full model, except that we remove the language-conditioned message passing module described in section 3.2.2. That is, we follow all the steps in section 3.2.1 and obtain every object’s local embedding, then map new embedding back to their locations as stated in section 3.2.3. The results in Table 3 indicate that language-conditioned message passing does help achieve higher exact match accuracy in many test splits, though it sometimes hurts the performance on split F. We conclude that the model is getting better at understanding object-related com-

mands (“pull” moves the object), sacrificing some ability to discover the meaning of easy-to-translate adverbs that are irrelevant to the interaction with objects (“while spinning” only describes the behavior of agent with no impact on the scene).

**Failure on Split B.** Here we analyze a failure case to understand why split B is notably difficult for our model. Figure 4 demonstrates an example that leads to both models’ failure. The attention scores indicate that the model has identified the correct target position, but does not know the correct action sequence to get there. The LSTM decoder cannot generalize the meaning of action tokens that direct the agent towards an unseen direction. We can observe from our model’s output prediction that, even if it manages to correctly predict the first few steps (“turn left turn left walk”), it quickly gets lost and fails to navigate to the target location. The model only observes the initial world state and the command, then generates a sequence of actions toward the target. In other words, it is blindly generating the action sequence with only a static image of the agent and the target’s location, not really modeling the movement of the agent. However, humans usually do not handle navigation in a novel direction in this way. Instead, they will first turn to the correct direction, and transform the novel task into a familiar task (“walk southwest is equivalent to turn southwest then walk the same as you walk north”). This naturally requires a change of perspective and conditioning on the agent’s previous action. A possible improvement is to introduce clues to inform the model of possible changes in its view as it takes actions.

## 5 Conclusion and Future Work

In this paper, we proposed a language-conditioned message passing model for a grounded language navigation task that can dynamically extract contextualized embeddings based on input command sentences, and can be trained end-to-end with the downstream action-sequence decoder. We showed that obtaining such contextualized embeddings improves performance on a recently introduced challenge problem, gSCAN, significantly outperforming the state-of-the-art across several test splits designed to test a model’s ability to represent novel concept compositions and achieve systematic generalization.

Nonetheless, our model’s fairly poor performance on split B and F shows that challenges still

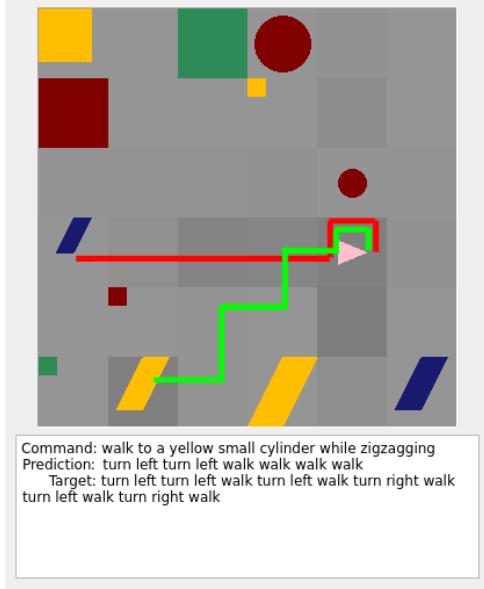


Figure 4: Failure case on split B, prediction and attention scores were generated by our model.

remain. As explained in the discussion section, our model is falling short of estimating the effect of each action on the agent’s state. An alternative view of this problem is as a reinforcement learning task with sparse reward. Sample-efficient model-based reinforcement learning (Buckman et al., 2018) could then be used, and its natural ability to explicitly model environment change should improve performance on this task.

It would also be beneficial to visualize the dynamically generated edge weights during message passing to have a more intuitive understanding of what contextual information is integrated during the message passing phase. Currently, we consider all objects appearing on the grid, including the agent, as homogeneous nodes during message passing, and all edges in the message passing graph are modelled in the same way. However, intuitively, we should model the relation between different types of objects differently. For example, the relation between the agent and the target object of pulling might be different from the relation between two objects on the grid. Inspired by Bahdanau et al. (2018), it would be interesting to try modeling different edge types explicitly with neural modules, and perform type-specific message passing to obtain better contextualized embeddings.

## References

- Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018a. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6077–6086.
- Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. 2018b. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3674–3683.
- Jacob Andreas. 2020. [Good-enough compositional data augmentation](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7556–7566, Online. Association for Computational Linguistics.
- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2016. [Neural machine translation by jointly learning to align and translate](#).
- Dzmitry Bahdanau, Shikhar Murty, Michael Noukhovitch, Thien Huu Nguyen, Harm de Vries, and Aaron C. Courville. 2018. [Systematic generalization: What is required and can it be learned?](#) *CoRR*, abs/1811.12889.
- Jacob Buckman, Danijar Hafner, George Tucker, Eugene Brevdo, and Honglak Lee. 2018. Sample-efficient reinforcement learning with stochastic ensemble value expansion. In *Advances in Neural Information Processing Systems*, pages 8224–8234.
- Maxime Chevalier-Boisvert, Dzmitry Bahdanau, Salem Lahou, Lucas Willems, Chitwan Saharia, Thien Huu Nguyen, and Yoshua Bengio. 2019. [Babyai: A platform to study the sample efficiency of grounded language learning](#).
- Jerry A Fodor, Zenon W Pylyshyn, et al. 1988. Connectionism and cognitive architecture: A critical analysis. *Cognition*, 28(1-2):3–71.
- Jonathan Gordon, David Lopez-Paz, Marco Baroni, and Diane Bouchacourt. 2019. Permutation equivariant models for compositional generalization in language. In *International Conference on Learning Representations*.
- Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. [Annotation artifacts in natural language inference data](#). In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

- Christina Heinze-Deml and Diane Bouchacourt. 2020. [Think before you act: A simple baseline for compositional generalization](#).
- Felix Hill, Andrew Lampinen, Rosalia Schneider, Stephen Clark, Matthew Botvinick, James L. McClelland, and Adam Santoro. 2020. [Environmental drivers of systematicity and generalization in a situated agent](#).
- Ronghang Hu, Jacob Andreas, Trevor Darrell, and Kate Saenko. 2018. Explainable neural computation via stack neural module networks. In *Proceedings of the European conference on computer vision (ECCV)*, pages 53–69.
- Ronghang Hu, Anna Rohrbach, Trevor Darrell, and Kate Saenko. 2019. [Language-conditioned graph networks for relational reasoning](#). 2019 IEEE/CVF International Conference on Computer Vision (ICCV).
- Drew Arad Hudson and Christopher D. Manning. 2018. [Compositional attention networks for machine reasoning](#). In *International Conference on Learning Representations*.
- Dieuwke Hupkes, Verna Dankers, Mathijs Mul, and Elia Bruni. 2020. Compositionality decomposed: How do neural networks generalise? *Journal of Artificial Intelligence Research*, 67:757–795.
- Robin Jia and Percy Liang. 2017. [Adversarial examples for evaluating reading comprehension systems](#). In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark. Association for Computational Linguistics.
- Hans Kamp and Barbara Partee. 1995. Prototype theory and compositionality. *Cognition*, 57(2):129–191.
- Yen-Ling Kuo, Boris Katz, and Andrei Barbu. 2020. [Compositional networks enable systematic generalization for grounded language understanding](#).
- Brenden M. Lake and Marco Baroni. 2017. [Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks](#).
- Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch.
- Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. 2018. Film: Visual reasoning with a general conditioning layer. In *Thirty-Second AAAI Conference on Artificial Intelligence*.
- Laura Ruis, Jacob Andreas, Marco Baroni, Diane Bouchacourt, and Brenden M. Lake. 2020. [A benchmark for systematic generalization in grounded language understanding](#).
- Minjie Wang, Da Zheng, Zihao Ye, Quan Gan, Mufei Li, Xiang Song, Jinjing Zhou, Chao Ma, Lingfan Yu, Yu Gai, Tianjun Xiao, Tong He, George Karypis, Jinyang Li, and Zheng Zhang. 2020. [Deep graph library: A graph-centric, highly-performant package for graph neural networks](#).
- Ziyun Wang and Brenden M. Lake. 2019. [Modeling question asking using neural program generation](#).

## A Appendix

### A.1 Implementation Details

Our implementation is based on the gSCAN dataset used by the Ruis et al. (2020) and the world size is  $d = 6$ .

For equation 1, each token is embedded to a randomly initialized vector of size 32, and the hidden size of the encoder BiLSTM is 32.

For equation 9, we use three convolutional networks with kernel size  $k = 1, 5, 7$  and padding size  $\lfloor \frac{k}{2} \rfloor = 0, 2, 3$  to ensure that the resulting dimensionality is synchronized with input. They share the same filter size of 64. The concatenation of  $H_i^s$  is also flattened to the shape of  $36 \times 192$ .

Table 4 presents the shapes of other trainable parameters mentioned in section 3. We simply set  $d_{cmd} = d_h = d_{loc} = d_m = d_s = d_{ctx} = 64$ .

Parameter	Shape
$W_1$	$1 \times d_{cmd}$
$W_2, W_3$	$d_{cmd} \times d_{cmd}$
$W_4$	$d_h \times d_{loc}$
$W_5$	$d_h \times d_{ctx}$
$W_6, W_7$	$d_h \times (d_{loc} + d_{ctx} + d_h)$
$W_8$	$d_h \times d_s$
$W_9$	$d_m \times (d_{loc} + d_{ctx} + d_h)$
$W_{10}$	$d_m \times d_s$
$W_{11}$	$d_{ctx} \times d_{ctx}$
$W_{12}$	$d_h \times (d_{loc} + d_{ctx})$

Table 4: Parameter Shapes

### A.2 Example Visualization

Here we present more examples demonstrating our model’s strengths and weaknesses. Figures 5 - 8 are cases where our model’s prediction exactly matches the target while the baseline’s does not. Some of the common failures of our model are illustrated in Figures 9 - 11.

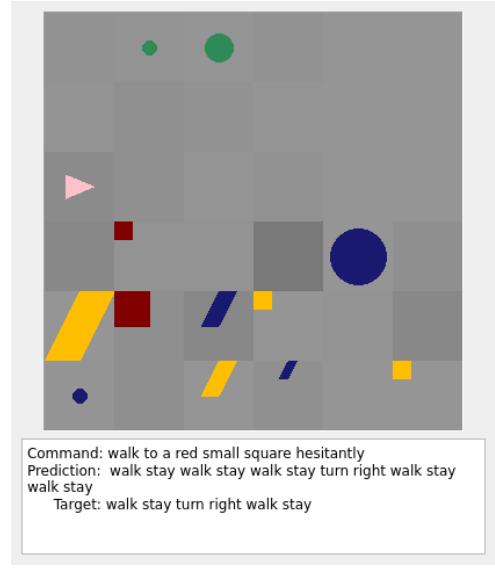


Figure 5: Baseline incorrectly picked a yellow square as the target.

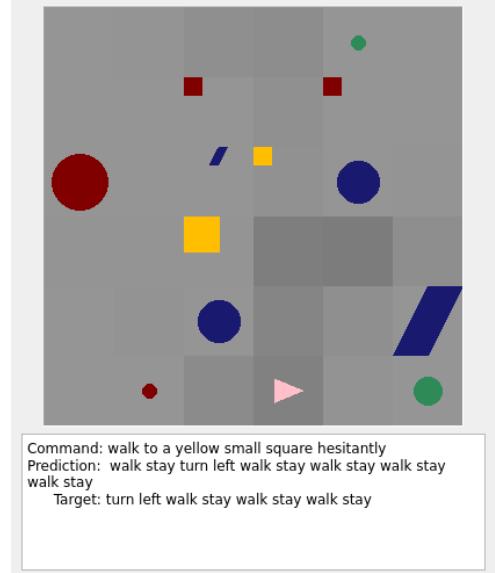


Figure 6: Baseline incorrectly picked a red square as the target.

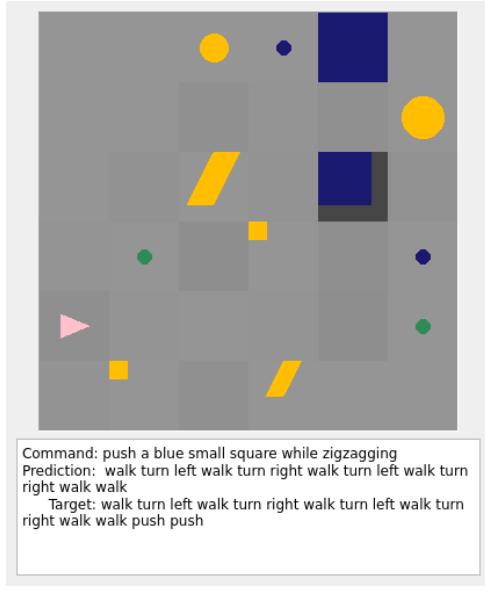


Figure 7: Baseline falsely predicted the consequential interaction and decided not to push.



Figure 9: Getting lost along a long sequence: Our model fails when the target sequence repeats the same actions several times.

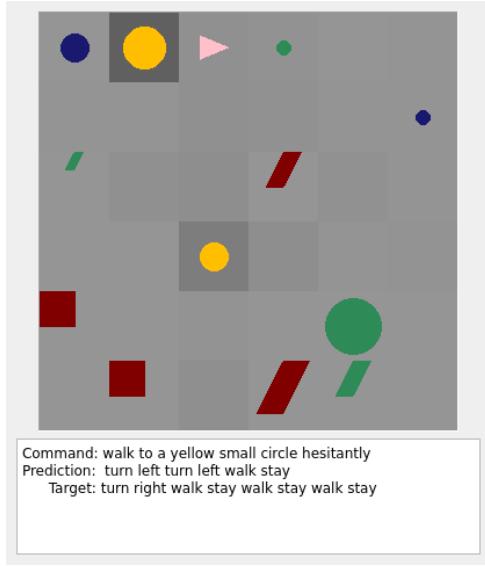


Figure 8: Baseline incorrectly picked the bigger circle instead of the smaller one.

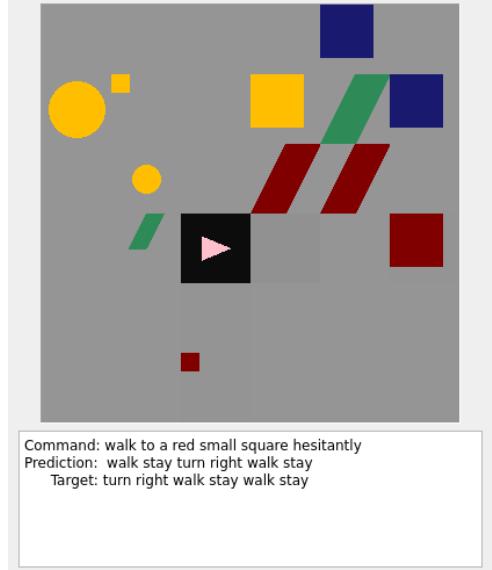


Figure 10: Incorrect path plan: Our model generates the path plan in a partially-reversed order.

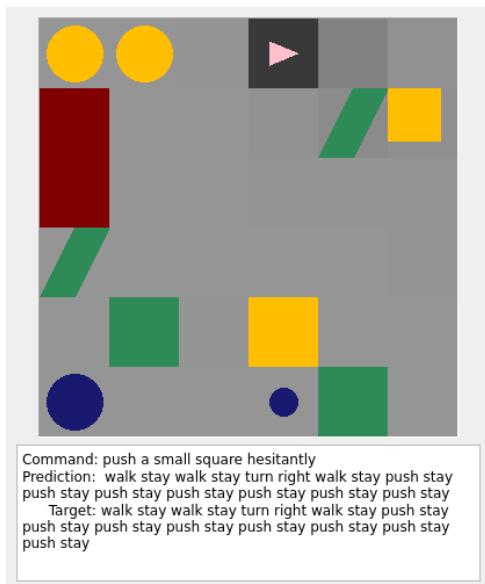


Figure 11: Early stop before reaching boundary: Our model stops pushing when the target object is next to the boundary grid.

# Are scene graphs good enough to improve Image Captioning?

Victor Milewski<sup>1</sup> Marie-Francine Moens<sup>1</sup> Iacer Calixto<sup>2,3</sup>

<sup>1</sup>KU Leuven <sup>2</sup>New York University <sup>3</sup>ILLC, University of Amsterdam  
{victor.milewski,sien.moens}@cs.kuleuven.be  
iacer.calixto@nyu.edu

## Abstract

Many top-performing image captioning models rely solely on object features computed with an object detection model to generate image descriptions. However, recent studies propose to directly use scene graphs to introduce information about object relations into captioning, hoping to better describe interactions between objects. In this work, we thoroughly investigate the use of scene graphs in image captioning. We empirically study whether using additional scene graph encoders can lead to better image descriptions and propose a *conditional* graph attention network (C-GAT), where the image captioning decoder state is used to condition the graph updates. Finally, we determine to what extent noise in the predicted scene graphs influence caption quality. Overall, we find no significant difference between models that use scene graph features and models that only use object detection features across different captioning metrics, which suggests that existing scene graph generation models are still too noisy to be useful in image captioning. Moreover, although the quality of predicted scene graphs is very low in general, when using high quality scene graphs we obtain gains of up to 3.3 CIDEr compared to a strong Bottom-Up Top-Down baseline.<sup>1</sup>

## 1 Introduction

Scene understanding is a complex and intricate activity which humans perform effortlessly but that computational models still struggle with. An important backbone of scene understanding is being able to detect objects and relations between objects in an image, and scene graphs (Johnson et al., 2015; Anderson et al., 2016) are a closely related data

structure that explicitly annotates an image with its objects and relations in context. Scene graphs can be used to improve important visual tasks that require scene understanding, e.g. image indexing and search (Johnson et al., 2015) or scene construction and generation (Johnson et al., 2017, 2018), and there is evidence that they can also be used to improve image captioning (Yang et al., 2019; Li and Jiang, 2019). However, the *de facto* standard in top-performing image captioning models to date use strong object features only, e.g. obtained with a pretrained Faster R-CNN (Ren et al., 2015), and no explicit relation information (Anderson et al., 2018; Lu et al., 2018; Yu et al., 2019a).

One possible explanation to this observation is that by using detected objects we already capture the more important information that characterises a scene, and that relation information is already *implicitly* learned in such models. Another explanation is that relations are simply not as important as we hypothesise and that we gain no valuable extra information by adding them. In this work, we investigate these empirical observations in more detail and strive to answer the following research questions: (i) Can we improve image captioning by explicitly supervising a model with information about object relations? (ii) How does the content of the captions improve when utilising scene graphs? (iii) How does scene graph quality impact the quality of the captions?

The most recent best-performing image captioning models make use of the Transformer architecture (Vaswani et al., 2017; Li et al., 2019; Yu et al., 2019b). However, in this paper we build upon the influential Bottom-Up Top-Down architecture (Anderson et al., 2018) which uses LSTMs, and since we want to measure to what extent scene graphs are helpful or not, we remove any “extras” to make model comparison easier, e.g. reinforcement learning step after cross-entropy training, ensembling at

<sup>1</sup>We open source the codebase to reproduce all our experiments in <https://github.com/iacer-calixto/butd-image-captioning>.

inference time, etc.

Scene graph generation (SGG) is the task where given an image a model predicts a graph with its objects and their relations. We use a pretrained SGG model (Xu et al., 2017) to obtain and inject explicit relation information into image captioning, and investigate different image captioning model architectures that incorporate object and relation features, similarly to Li and Jiang (2019); Wang et al. (2019). We propose an extension to graph attention networks (Veličković et al., 2018) which we call conditional graph attention (C-GAT), where we condition the updates of the scene graph features on the current image captioning decoder state. Finally, we conduct an in-depth analysis of the captions produced by different models and determine if scene graphs actually improve the content of the captions. Our approach is illustrated in Figure 1.

Our main contributions are:

- We investigate different graph-based architectures to fuse object and relation information derived from scene graph generation models in the context of image captioning.
- We introduce conditional graph attention networks to condition scene graph updates on the current state of an image captioning decoder and find that it leads to improvements of up to 0.8 CIDEr.
- We compare the quality of the generated scene graphs and the quality of the corresponding captions and find that by using high quality scene graphs we can improve captions quality by up to 3.3 CIDEr.
- We systematically analyse captions generated by standard image captioning models and by models with access to scene graphs using SPICE scores for objects and relations (Anderson et al., 2016) and find that when using scene graphs there is an increase of 0.4 F1 for relations and decrease of 0.1 F1 for objects.

## 2 Background

### 2.1 Object Detection

Object detection is a task where given an input image the goal is to locate and label all its objects. The **Faster R-CNN**, which builds on the R-CNN and Fast R-CNN (Girshick et al., 2014; Girshick, 2015), is a widely adopted model proposed for object detection (Ren et al., 2015). It uses a pretrained convolutional neural network (CNN) as a backbone to extract feature maps for an input image. A region

proposal network (RPN) uses these feature maps to propose a set of regions with a high likelihood of containing an object. For each region, a feature vector is generated using the feature map, which is then passed to an object classification layer. In our experiments, we use the Faster R-CNN with a ResNet-101 backbone (He et al., 2016).

### 2.2 Graph Neural Networks

Graph neural networks (GNNs; Battaglia et al., 2018) are neural architectures designed to operate on arbitrarily structured graphs  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ , where  $\mathcal{V}$  and  $\mathcal{E}$  are the set of vertices and edges in  $\mathcal{G}$ , respectively. In GNNs, representations for a vertex  $v \in \mathcal{V}$  are computed by using information from neighbouring vertices  $\mathcal{N}(v)$  which are defined to include all vertices connected through an edge. In this work, we use a neighbourhood  $\mathcal{N}(v_1)$  that contains vertices  $v_2$  connected through *incoming* edges, i.e.  $v_2 \rightarrow v_1 \in \mathcal{E}$ .

**Graph Attention Networks** Graph attention networks (GATs; Veličković et al., 2018) combine features from neighbour vertices  $\mathcal{N}(v_1)$  through an attention mechanism (Bahdanau et al., 2014) to generate representations for vertex  $v_1$ . Vertex  $v_1$ 's state  $v_1^{t-1}$  at time step  $t - 1$  is used as the query to soft-select the information from neighbours relevant to its updated state  $v_1^t$ .

### 2.3 Scene Graphs

Scene graphs consist of a data structure devised to annotate an image with its objects and the existing relations between objects and were first introduced for image retrieval (Johnson et al., 2015).

We consider scene graphs  $\mathcal{G}$  for an image with two types of *vertices*: objects and relations.<sup>2</sup> Object vertices describe the different objects in the image, and relation vertices describe how different objects interact with each other. This gives us the following rules for edges  $\mathcal{E}$ : (i) All existing edges are between an object vertex and a relation vertex; (ii) If an object  $o_1$  is connected to another object  $o_2$  via a relation vertex  $r_3$ , then vertex  $r_3$  has *only* two connected edges: one incoming from  $o_1$  and one outgoing to  $o_2$ . Finally, object (relation) vertices are also associated to an object (relation) label.

**Scene Graph Generation** Scene graph generation (SGG) was introduced by Xu et al. (2017) and

<sup>2</sup>Attributes are also originally present in scene graphs as vertices, but we do not use them.

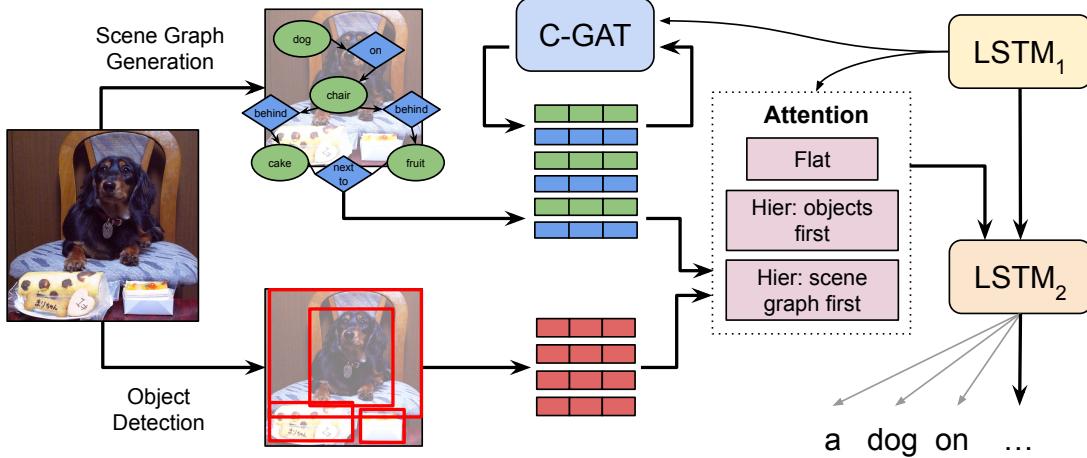


Figure 1: We use object features from an object detection model and scene graph features from a scene graph generation model in image captioning. We use conditional graph attention (C-GAT) to encode scene graph features, and flat vs. hierarchical attention mechanisms are used to incorporate both feature sets into an LSTM decoder.

has since received growing attention (Zellers et al., 2018; Li et al., 2018; Yang et al., 2018; Knyazev et al., 2020). One can compare it to object detection (Section 2.1), where instead of only predicting objects a model must additionally predict which objects have relations and what are these relations. This similarity makes it natural that SGG models build on object detection architectures. Most SGG models use a pretrained Faster R-CNN or similar architecture to predict objects and have an additional component to predict relations for pairs of objects. In addition to the original object loss components in the Faster R-CNN, they include a mechanism to update object feature representations using neighbourhood information, and a component to predict relations and their label.

**Iterative Message Passing** The Iterative Message Passing SGG model (Xu et al., 2017) keeps two sets of states, i.e. for object vertices and relation vertices. The object vertices are initialised directly from Faster R-CNN features, while a relation vertex is computed by the box union of each of its two objects boxes, which is encoded with the Faster R-CNN to obtain a relation vector.

Hidden states in each set are updated using an attention mechanism over neighbour vertices, i.e. objects are informed by all connected relation vertices, and relations are informed by the two objects it links. Since there are two sets of states it is easy to efficiently send messages from one set to the other by the means of an adjacency matrix. This procedure is repeated for  $k$  iterations, and Xu et al. (2017) found that  $k = 2$  gives optimal results.

**Relation proposal network (RelPN)** Xu et al. (2017) first proposed to build a fully connected graph connecting all object pairs and scoring relations between all possible object pairs; however, this model is expensive and grows exponentially with the number of objects. Yang et al. (2018) introduced a relation proposal network (RelPN), which works similarly to an object detection RPN but that selectively proposes relations between pairs of objects. In all our experiments, we use the Iterative Message Passing model trained using a RelPN.

### 3 Conditional Graph Attention (C-GAT)

Standard graph neural architectures encode information about neighbour nodes  $\mathcal{N}(v)$  into representations of node  $v \in \mathcal{V}$ . Therefore, these GNNs are *contextual* because they encode *graph-internal* context.

We propose the **conditional graph attention (C-GAT)** architecture, a novel extension for graph attention networks (Veličković et al., 2018).<sup>3</sup> Our goal is to make these networks *conditional* in addition to contextual. By conditional we mean that a C-GAT layer is conditioned on *external* context, e.g. a vector representing knowledge that is not part of the original input graph.

Our motivation is that when using graph-based inputs such as a scene graph, a C-GAT layer allows us to condition the message propagation between connected nodes in the graph on the current state of the model, e.g. on the decoder state in the

<sup>3</sup>This architecture is novel to the best of our knowledge.

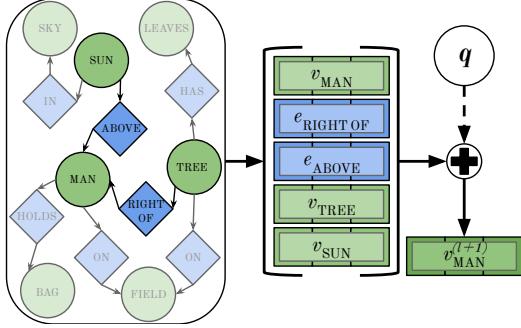


Figure 2: C-GAT layer where we illustrate the update of vertex  $\text{MAN}$  by combining the features of all incoming relations (and objects) through an attention mechanism. The attention scores are conditioned on the external query vector  $q$ .

captioning decoder in Figure 1. Whereas a standard GAT layer contextually updates object hidden states, it cannot condition on context outside the scene graph.<sup>4</sup> With a C-GAT layer, we provide a mechanism for the model to learn to update object hidden states *in the context of the current state of the decoder language model*, which we expect to lead to better contextual features.

In Figure 2, a C-GAT layer is applied to an input scene graph  $\mathcal{G}$  and conditioned on a query vector  $q$ , i.e. the decoder state. We illustrate the update of vertex  $v_{\text{man}} \in \mathcal{V}$  using features from its neighbourhood  $\mathcal{N}(v_{\text{man}})$ . The self-relation is assumed to always be present and for readability is not shown. Neighbour nodes’ features are combined with an MLP attention mechanism (Bahdanau et al., 2014) and scores are computed using query  $q$ .

As described in Section 2.3, neighbours of an object vertex  $v_i \in \mathcal{V}$  in a scene graph only include relation vertices. To include neighbour object features as well as relation features, we collect features for all  $v_j \in \mathcal{N}_{\text{obj}}(v_i)$ , defined as nodes accessible by all relation vertices  $v_r$  such that  $\{v_i \leftarrow v_r, v_r \leftarrow v_j\} \in \mathcal{E}$ .

## 4 Model Setup

In this section, we first introduce image captioning models that do not explicitly use relation features (Section 4.1) and contrast them with those that use explicit relation features (Section 4.2).

### 4.1 Baseline Image Captioning (IC)

**Bottom-Up Top-Down (BUTD)** The bottom-up top-down (BUTD; Anderson et al., 2018) model

<sup>4</sup>This is generally true for standard GNN architectures and not just GATs.

consists of a Faster R-CNN image encoder (Ren et al., 2015) that computes object proposal features for an input image, and a 2-layer LSTM language model decoder with a MLP attention mechanism over the object features that generates a caption for the image (Hochreiter and Schmidhuber, 1997; Bahdanau et al., 2014). We denote the set of object features  $\mathbf{X} \in \mathbb{R}^{n \times d}$ , where  $n$  is the number of objects in the image and  $d$  the features dimensionality. The 2-layer LSTM is designed so that the first layer is used to compute an attention over the image features and the second layer is used to generate the captions’ tokens. LSTM states at time step  $t$  are denoted as  $\mathbf{h}_1^{(t)}$  and  $\mathbf{h}_2^{(t)}$  for layer 1 and 2, respectively. The hidden state of  $\text{LSTM}_1$  is used to derive an attention over image features:

$$\mathbf{x}^{(t)} = \text{Att}(\mathbf{X}, \mathbf{h}_1^{(t)}), \quad (1)$$

where  $\mathbf{x}^{(t)} \in \mathbb{R}^d$  is the output of the attention layer and denotes the image features used at time step  $t$ . Update rules for each LSTM layer are defined by:

$$\mathbf{h}_1^{(t)} = \text{LSTM}_1([\mathbf{h}_2^{(t-1)}; \mathbf{w}^{(t-1)}; \bar{\mathbf{X}}], \quad \mathbf{h}_1^{(t-1)}), \quad (2)$$

$$\mathbf{h}_2^{(t)} = \text{LSTM}_2([\mathbf{h}_1^{(t)}; \mathbf{x}^{(t)}], \quad \mathbf{h}_2^{(t-1)}), \quad (3)$$

where  $\mathbf{w}^{(t-1)}$  is the embedding of the previously generated word, and  $\bar{\mathbf{X}} \in \mathbb{R}^d$  are the mean image features.

Next word probabilities are computed using a softmax over the vocabulary and parameterised by a linear projection of the hidden state of  $\text{LSTM}_2$ :  $p(w^{(t)} = k | \mathbf{w}^{(1):(t-1)}) \propto \exp(\mathbf{W}\mathbf{h}_2^{(t)})$ .

### 4.2 Relation-aware Image Captioning (RIC)

We now describe models that incorporate explicit relation information into image captioning by using scene graphs as additional inputs.

We use the pretrained Iterative Message Passing model with a relation proposal network (Xu et al., 2017; Yang et al., 2018) to obtain scene graph features for all images. Scene graph features for an image are denoted  $\mathbf{Y} \in \mathbb{R}^{(o+r) \times k}$ , where  $o$  is the number of objects,  $r$  the number of relations between objects, and  $k$  is the object/relation feature dimensionality.

We follow Wang et al. (2019) who have found that only using scene graph features led to poor results compared to using Faster R-CNN features only. Therefore, we propose to integrate scene

graph features  $\mathbf{Y}$  and Faster R-CNN object features  $\mathbf{X}$  by experimenting with (i) using  $\mathbf{Y}$  directly, applying a GAT layer on  $\mathbf{Y}$ , or applying a C-GAT layer on  $\mathbf{Y}$  prior to feeding scene graph features into the decoder, and (ii) using a flat attention mechanism versus a hierarchical attention mechanism.

**GAT over Scene Graphs** We propose a model that encodes the scene graph features  $\mathbf{Y}$  with a standard GAT layer prior to using them in  $\text{LSTM}_2$  in the decoder.

**C-GAT over Scene Graphs** In this setup, we apply a C-GAT layer on scene graph features  $\mathbf{Y}$  using the current decoder state  $\mathbf{h}_1^{(t)}$  from  $\text{LSTM}_1$  as the external context, and use the output of the C-GAT layer in  $\text{LSTM}_2$  in the decoder.

**Flat Attention** The flat attention (FA) consists of two separate attention heads, one over scene graph features  $\mathbf{Y}$  and the other over Faster R-CNN features  $\mathbf{X}$ . We use two standard MLP attention mechanisms (Bahdanau et al., 2014), each using the hidden state from  $\text{LSTM}_1$  as the query:

$$\begin{aligned}\mathbf{x}^{(t)} &= \text{Att}_x(\mathbf{X}, \mathbf{h}_1^{(t)}), \\ \mathbf{y}^{(t)} &= \text{Att}_y(\mathbf{Y}, \mathbf{h}_1^{(t)}).\end{aligned}$$

Each LSTM layer is now defined as follows:

$$\begin{aligned}\mathbf{h}_1^{(t)} &= \text{LSTM}_1([\mathbf{h}_2^{(t-1)}; \mathbf{w}^{(t-1)}; \bar{\mathbf{X}}; \bar{\mathbf{Y}}], \mathbf{h}_1^{(t-1)}), \\ \mathbf{h}_2^{(t)} &= \text{LSTM}_2([\mathbf{h}_1^{(t)}; \mathbf{x}^{(t)}; \mathbf{y}^{(t)}], \mathbf{h}_2^{(t-1)}),\end{aligned}\quad (4)$$

where  $\mathbf{x}^{(t)}$  and  $\mathbf{y}^{(t)}$  are computed by the two attention heads  $\text{Att}_x$  and  $\text{Att}_y$ , respectively, and  $\bar{\mathbf{Y}}$  denote the mean scene graph features.

**Hierarchical Attention** In a hierarchical attention (HA) mechanism the output of the first attention head is used as input to derive the attention of the second head. We again have two sets of inputs, scene graph features  $\mathbf{Y}$  and Faster R-CNN object features  $\mathbf{X}$ . We experiment first using  $\mathbf{Y}$  as input to the first head, and its output  $\mathbf{y}^{(t)}$  as additional input to the second head:

$$\begin{aligned}\mathbf{y}^{(t)} &= \text{Att}_y(\mathbf{Y}, \mathbf{h}_1^{(t)}), \\ \mathbf{x}^{(t)} &= \text{Att}_x(\mathbf{X}, [\mathbf{h}_1^{(t)}; \mathbf{y}^{(t)}]).\end{aligned}\quad (5)$$

This setup is similar to the cascade attention from Wang et al. (2019). We also try using  $\mathbf{X}$

as input to the first head, and the first head’s output  $\mathbf{x}^{(t)}$  as additional input to the second head:

$$\begin{aligned}\mathbf{x}^{(t)} &= \text{Att}_x(\mathbf{X}, \mathbf{h}_1^{(t)}). \\ \mathbf{y}^{(t)} &= \text{Att}_y(\mathbf{Y}, [\mathbf{h}_1^{(t)}; \mathbf{x}^{(t)}]).\end{aligned}\quad (6)$$

In both cases, the hidden states for  $\text{LSTM}_1$  and  $\text{LSTM}_2$  are computed as in Equation 4.

## 5 Experimental Setup and Results

We compare our models with the following external baselines with no access to scene graphs: (1) the adaptive attention model **Add-Att** which determines at each decoder time step how much of the visual features should be used (Lu et al., 2017); (2) the Neural Baby Talk model **NBT** generates a sentence with gaps and fills the gaps using detected object labels (Lu et al., 2018); (3) and the **BUTD** model (Anderson et al., 2018) described in Section 4.1.

We also compare with the following baselines that use scene graphs: (1) The “Know more, say less” model **KMSL** extracts features for objects and relations based on the scene graph, which are passed through two attention heads and finally combined using a flat attention head (Li and Jiang, 2019); and (2) the **Cascade** model (Wang et al., 2019) which is similar to our hierarchical attention model with a GAT layer, but that instead uses a relational graph convolutional network (Marcheggiani et al., 2017).

We do not discuss model variants/results that are trained with an additional reinforcement learning step (Rennie et al., 2017; Yang et al., 2019) and only compare single model results, since training and performing inference with such models is very costly and orthogonal to our research questions.

Our proposed models are: flat attention (**FA**), hierarchical attention with scene graph first (**HA-SG**) following Equation 5, hierarchical attention with objects detected first (**HA-IM**) following Equation 6, HA-SG with graph attention network (**HA-SG+GAT**), and HA-SG with conditional graph attention (**HA-SG+C-GAT**). We choose the last two variants to extend HA-SG following the setup used by (Wang et al., 2019).

We evaluate captions generated by different models by investigating their SPICE scores (Anderson et al., 2016), i.e. an F1 based semantic captioning evaluation metric computed over scene graphs. It uses the semantic structure of the scene graph to

	B4	C	R	S
Add-Att* <sup>†</sup>	33.2	108.5	—	—
NBT <sup>†</sup>	34.7	107.2	—	20.1
BUTD <sup>†</sup>	36.2	113.5	56.4	20.3
BUTD	34.8	109.2	55.7	20.0
Cascade <sup>†</sup>	34.1	108.6	<u>55.9</u>	<b>20.3</b>
KMSL <sup>†</sup>	33.8	<b>110.3</b>	54.9	19.8
FA	33.7	102.5	54.7	18.8
HA-IM	<b>35.7</b>	<u>109.9</u>	<u>55.9</u>	<u>19.9</u>
HA-SG	35.0	109.1	55.7	19.8
+ GAT	34.7	106.4	55.4	19.4
+ C-GAT	<u>35.5</u>	<u>109.9</u>	<b>56.0</b>	19.8

Table 1: Results on the MSCOCO test set, with models selected on the validation set (*karpathy* splits). Models in the upper section do not use scene graphs, while those in the bottom section do. All models are trained to convergence for a maximum of 50 epochs. Metrics reported are: BLEU-4 (B4), CIDEr (C), ROUGE-L (R), and SPICE (S). See Section 5 for details on all models and acronyms. We bold-face the best and underscore the second-best scores per metric (models that use scene graph). \* Model uses features from last convolutional layer in CNN, i.e. no Faster R-CNN features.

<sup>†</sup> Results reported in the authors’ original papers.

compute scores over several dimensions (object, relation, attribute, colour, count, and size).

We use the MSCOCO *karpathy split* (Lin et al., 2014; Karpathy and Fei-Fei, 2015) which has 5k images each in validation and test sets, and we use the remaining 113k images for training. We build a vocabulary based on all words in the train split that occur at least 5 times. We use MSCOCO evaluation scripts (Lin et al., 2014) and report BLEU4 (B4; Papineni et al., 2002), CIDEr (C; Vedantam et al., 2015), ROUGE-L (R; Lin, 2004), and SPICE (S; Anderson et al., 2016). See Appendix A for extra information on our implementation and training procedures.

## 5.1 Image Captioning without Relational Features

Our re-implementation of the BUTD baseline scores slightly worse compared to the results reported by Anderson et al. (2018). This difference can be attributed to the Faster R-CNN features used, i.e. we always use 36 objects per image whereas Anderson et al. (2018) use a variable number of objects per image (i.e. 10 to 100), and there are other smaller differences in their training procedure.

Since all our models use these settings, in further experiments we compare to our implementation of the BUTD baseline.

## 5.2 Image Captioning with relational features

We notice that the KMSL model by Li and Jiang (2019) slightly outperforms the other models according to CIDEr, while it performs worse in all other metrics. Li and Jiang (2019) found performance increases when restricting the number of relations and report scores using this restriction, whereas we decided to use the full set of relations to test the effect of scene graph quality (see Section 5.4). Furthermore, the features used in the KMSL model are not directly extracted from the SGG model as is the case for the other models, but an additional architecture is used for computing stronger features.

**Flat vs. Hierarchical attention** According to Table 1, FA performs worse not only compared to HA models, but also compared to other baselines.

The HA model using Faster R-CNN object features in the first head, i.e. HA-IM setup, performs better than using the scene graph features first, i.e. HA-SG setup. We hypothesise that this difference comes from the additional guidance from  $\mathbf{x}^{(t)}$  helping with a better attention selection over possibly more noisy features present in  $\mathbf{Y}$ .

**Additional GNN updates** Directly using a GAT layer over scene graph features negatively impacts model performance. Comparing these results to the related Cascade model from Wang et al. (2019), we hypothesise that the R-GCN architecture works better in this setting, although compared to other models it still has lower scores according to most metrics. The reason may be that the Cascade model by Wang et al. (2019) was undertrained or could have used better hyperparameters, as indicated by our BUTD baseline performing comparably or better than their strongest model.<sup>5</sup>

Combining a C-GAT layer on the decoder improves overall results according to most metrics, though by a small margin. This suggests that using additional GNNs in the context of image captioning have a positive effect. Furthermore, graph features learned using C-GAT always outperform standard GAT, which coincides with our intuition that taking

<sup>5</sup>Our BUTD baseline scores 109.2 CIDEr, whereas their best model achieves 108.6 CIDEr.

	All	Obj	Rel
BUTD	<b>19.8</b>	<b>36.0</b>	5.2
FA	18.5	34.7	5.0
HA-IM	<u>19.5</u>	<u>35.9</u>	5.2
HA-SG	<u>19.5</u>	<u>35.9</u>	5.3
+ GAT	19.2	35.5	<b>5.6</b>
+ C-GAT	19.4	35.8	5.3

Table 2: Breakdown of overall SPICE scores (All) into object (Obj) and relation (Rel) F1 scores. See Section 5 for details on all models and acronyms. We bold-face the best overall scores and underline the best scores obtained by our models.

the current decoder hidden context into consideration can improve graph features.

### 5.3 SPICE breakdown

In our analysis, in addition to the overall SPICE F1 score for an entire caption, we break it down into scores over objects and over relations.<sup>6</sup> This allows us to investigate how models are better or worse on describing objects and relations independently. These results, computed for the validation split, are shown in Table 2.

When we look at individual scores for objects and relations, we notice a small and consistent gain in relation F1 by using scene graphs independently of the attention architecture or other design choices, but also observe lower object F1 scores with respect to the BUTD baseline. When object and relation scores are combined into a single F1 measure, it results in worse overall scores suggesting that the small increase in the relation scores is not sufficient to have a positive impact on captioning insofar.

### 5.4 Scene Graph Quality

Since scene graph features are generated with a pre-trained SGG model, we expect them to introduce a considerable amount of noise into the model. In this section, we investigate the effect that the quality of the scene graph has on the quality of captions.

**VG-COCO** In this set of experiments, we need images with *both* captions and scene graph annotations. Thus, we use a subset of MSCOCO which overlaps with Visual Genome (Krishna et al., 2017), using captions from the former and scene graphs

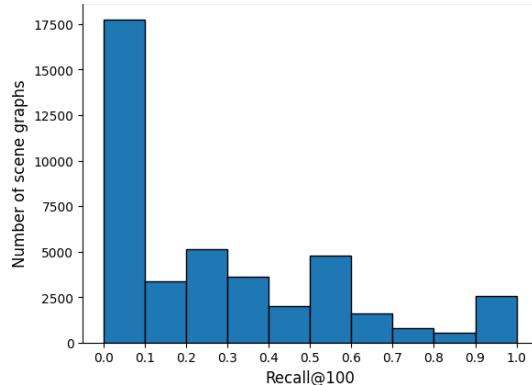


Figure 3: Distribution of the scores for the scene graphs in the validation split of the VG-COCO dataset.

from the latter. We refer to this dataset as VG-COCO, as similarly done by Li and Jiang (2019). We compute scores for each scene graph predicted by the Iterative Message Passing model using the common *SGDet recall@100* as defined by Yang et al. (2018). SGDet recall@100 is computed by using the 100 highest scoring triplets among all triplets predicted by the model,<sup>7</sup> and reporting the percentage of gold-standard triplets. The distribution of scores across images (Figure 3) shows that most scene graphs have extremely low scores close to zero, thus containing a lot of noise.

We separate images in the VG-COCO validation set in three groups: low ( $R < 33\%$ ), average ( $33\% \leq R < 67\%$ ), and high scoring graphs ( $67\% \leq R$ ), where  $R$  is SGDet recall@100. For each set of images in each of these groups, we compute captioning metrics and also report a SPICE breakdown in Table 3.

**Effect of scene graph quality** Due to the imbalance in scene graph quality, the *low*, *average*, and *high* quality subsets have around 1000, 500, and 200 images, respectively. By reporting results for the BUTD baseline, we show the performance a strong baseline obtains on the same set of images.

In Table 3b, scores across all metrics are similar and only model FA performs clearly worse than others. Though the BUTD baseline never performs best, it is often not more than a point behind the best performing model (except for CIDEr where it is 2.5 points lower compared to HA-IM).

When comparing Table 3b to Table 3c, we observe that all models tend to increase scores, and that BUTD tends to perform best overall. In Table 3d, we see an increase in the difference between

<sup>6</sup>The SPICE score also includes the components attribute, colour, count, and size, but we do not report them directly.

<sup>7</sup>A triplet is an object-predicate-subject phrase.

	SPICE			Captioning			SPICE			Captioning		
	All	Obj	Rel	B4	C	R	All	Obj	Rel	B4	C	R
BUTD	<b>19.8</b>	<u>36.0</u>	5.0	35.4	109.8	<b>56.0</b>	<u>19.5</u>	35.6	4.7	34.0	109.2	55.2
FA	18.5	34.9	4.8	33.2	103.6	54.8	18.2	34.9	4.4	32.4	102.7	54.3
HA-IM	19.6	<u>36.0</u>	5.2	35.0	<u>110.8</u>	55.7	<u>19.5</u>	<b>36.0</b>	<u>5.2</u>	34.3	<b>111.7</b>	<u>55.5</u>
HA-SG	<b>19.8</b>	<b>36.2</b>	<u>5.5</u>	<u>35.7</u>	<b>111.0</b>	<b>56.0</b>	<b>19.6</b>	<b>36.0</b>	5.0	34.6	<u>110.3</u>	55.4
+ GAT	19.4	35.5	<b>5.6</b>	35.0	108.3	55.6	<u>19.5</u>	35.7	<b>5.7</b>	<u>34.8</u>	109.5	55.4
+ C-GAT	19.5	35.8	5.2	<b>35.7</b>	110.4	<b>56.0</b>	<u>19.5</u>	35.7	5.1	<u>34.8</u>	109.9	<b>55.7</b>

(a) Full VG-COCO dataset	(b) Low VG-COCO dataset																																																																																																																																																																																																														
<table border="1"> <thead> <tr> <th rowspan="2"></th><th colspan="3">SPICE</th><th colspan="3">Captioning</th><th colspan="3">SPICE</th><th colspan="3">Captioning</th></tr> <tr> <th>All</th><th>Obj</th><th>Rel</th><th>B4</th><th>C</th><th>R</th><th>All</th><th>Obj</th><th>Rel</th><th>B4</th><th>C</th><th>R</th></tr> </thead> <tbody> <tr> <td>BUTD</td><td><b>20.5</b></td><td><b>37.0</b></td><td><u>5.5</u></td><td><b>38.4</b></td><td><b>117.5</b></td><td><b>57.1</b></td><td><b>20.9</b></td><td><u>36.8</u></td><td>5.1</td><td><u>37.2</u></td><td>126.5</td><td>57.0</td></tr> <tr> <td>FA</td><td>18.8</td><td>35.0</td><td>5.3</td><td>34.6</td><td>111.1</td><td>55.1</td><td>19.8</td><td>35.3</td><td>5.6</td><td>35.9</td><td>117.4</td><td>56.6</td></tr> <tr> <td>HA-IM</td><td><u>19.8</u></td><td><u>36.2</u></td><td>5.3</td><td>36.2</td><td>115.1</td><td>55.3</td><td>20.3</td><td>36.2</td><td>5.8</td><td>36.2</td><td>124.1</td><td>56.8</td></tr> <tr> <td>HA-SG</td><td>19.6</td><td>35.9</td><td>5.4</td><td>37.0</td><td>114.4</td><td>56.3</td><td><b>20.9</b></td><td><b>37.1</b></td><td><u>6.0</u></td><td><b>38.1</b></td><td><b>129.8</b></td><td><b>57.6</b></td></tr> <tr> <td>+ GAT</td><td>19.4</td><td>35.8</td><td><b>5.7</b></td><td>36.2</td><td>112.7</td><td>56.0</td><td>19.7</td><td>35.3</td><td>5.9</td><td>36.2</td><td>123.6</td><td><u>57.1</u></td></tr> <tr> <td>+ C-GAT</td><td>19.5</td><td>36.0</td><td>5.1</td><td><u>37.6</u></td><td><u>116.4</u></td><td><u>56.1</u></td><td>20.8</td><td>36.6</td><td><b>6.0</b></td><td><u>37.2</u></td><td><u>127.3</u></td><td>56.9</td></tr> </tbody> </table>		SPICE			Captioning			SPICE			Captioning			All	Obj	Rel	B4	C	R	All	Obj	Rel	B4	C	R	BUTD	<b>20.5</b>	<b>37.0</b>	<u>5.5</u>	<b>38.4</b>	<b>117.5</b>	<b>57.1</b>	<b>20.9</b>	<u>36.8</u>	5.1	<u>37.2</u>	126.5	57.0	FA	18.8	35.0	5.3	34.6	111.1	55.1	19.8	35.3	5.6	35.9	117.4	56.6	HA-IM	<u>19.8</u>	<u>36.2</u>	5.3	36.2	115.1	55.3	20.3	36.2	5.8	36.2	124.1	56.8	HA-SG	19.6	35.9	5.4	37.0	114.4	56.3	<b>20.9</b>	<b>37.1</b>	<u>6.0</u>	<b>38.1</b>	<b>129.8</b>	<b>57.6</b>	+ GAT	19.4	35.8	<b>5.7</b>	36.2	112.7	56.0	19.7	35.3	5.9	36.2	123.6	<u>57.1</u>	+ C-GAT	19.5	36.0	5.1	<u>37.6</u>	<u>116.4</u>	<u>56.1</u>	20.8	36.6	<b>6.0</b>	<u>37.2</u>	<u>127.3</u>	56.9	<table border="1"> <thead> <tr> <th rowspan="2"></th><th colspan="3">SPICE</th><th colspan="3">Captioning</th><th colspan="3">SPICE</th><th colspan="3">Captioning</th></tr> <tr> <th>All</th><th>Obj</th><th>Rel</th><th>B4</th><th>C</th><th>R</th><th>All</th><th>Obj</th><th>Rel</th><th>B4</th><th>C</th><th>R</th></tr> </thead> <tbody> <tr> <td>BUTD</td><td><b>20.5</b></td><td><b>37.0</b></td><td><u>5.5</u></td><td><b>38.4</b></td><td><b>117.5</b></td><td><b>57.1</b></td><td><b>20.9</b></td><td><u>36.8</u></td><td>5.1</td><td><u>37.2</u></td><td>126.5</td><td>57.0</td></tr> <tr> <td>FA</td><td>18.8</td><td>35.0</td><td>5.3</td><td>34.6</td><td>111.1</td><td>55.1</td><td>19.8</td><td>35.3</td><td>5.6</td><td>35.9</td><td>117.4</td><td>56.6</td></tr> <tr> <td>HA-IM</td><td><u>19.8</u></td><td><u>36.2</u></td><td>5.3</td><td>36.2</td><td>115.1</td><td>55.3</td><td>20.3</td><td>36.2</td><td>5.8</td><td>36.2</td><td>124.1</td><td>56.8</td></tr> <tr> <td>HA-SG</td><td>19.6</td><td>35.9</td><td>5.4</td><td>37.0</td><td>114.4</td><td>56.3</td><td><b>20.9</b></td><td><b>37.1</b></td><td><u>6.0</u></td><td><b>38.1</b></td><td><b>129.8</b></td><td><b>57.6</b></td></tr> <tr> <td>+ GAT</td><td>19.4</td><td>35.8</td><td><b>5.7</b></td><td>36.2</td><td>112.7</td><td>56.0</td><td>19.7</td><td>35.3</td><td>5.9</td><td>36.2</td><td>123.6</td><td><u>57.1</u></td></tr> <tr> <td>+ C-GAT</td><td>19.5</td><td>36.0</td><td>5.1</td><td><u>37.6</u></td><td><u>116.4</u></td><td><u>56.1</u></td><td>20.8</td><td>36.6</td><td><b>6.0</b></td><td><u>37.2</u></td><td><u>127.3</u></td><td>56.9</td></tr> </tbody> </table>		SPICE			Captioning			SPICE			Captioning			All	Obj	Rel	B4	C	R	All	Obj	Rel	B4	C	R	BUTD	<b>20.5</b>	<b>37.0</b>	<u>5.5</u>	<b>38.4</b>	<b>117.5</b>	<b>57.1</b>	<b>20.9</b>	<u>36.8</u>	5.1	<u>37.2</u>	126.5	57.0	FA	18.8	35.0	5.3	34.6	111.1	55.1	19.8	35.3	5.6	35.9	117.4	56.6	HA-IM	<u>19.8</u>	<u>36.2</u>	5.3	36.2	115.1	55.3	20.3	36.2	5.8	36.2	124.1	56.8	HA-SG	19.6	35.9	5.4	37.0	114.4	56.3	<b>20.9</b>	<b>37.1</b>	<u>6.0</u>	<b>38.1</b>	<b>129.8</b>	<b>57.6</b>	+ GAT	19.4	35.8	<b>5.7</b>	36.2	112.7	56.0	19.7	35.3	5.9	36.2	123.6	<u>57.1</u>	+ C-GAT	19.5	36.0	5.1	<u>37.6</u>	<u>116.4</u>	<u>56.1</u>	20.8	36.6	<b>6.0</b>	<u>37.2</u>	<u>127.3</u>	56.9
		SPICE			Captioning			SPICE			Captioning																																																																																																																																																																																																				
	All	Obj	Rel	B4	C	R	All	Obj	Rel	B4	C	R																																																																																																																																																																																																			
BUTD	<b>20.5</b>	<b>37.0</b>	<u>5.5</u>	<b>38.4</b>	<b>117.5</b>	<b>57.1</b>	<b>20.9</b>	<u>36.8</u>	5.1	<u>37.2</u>	126.5	57.0																																																																																																																																																																																																			
FA	18.8	35.0	5.3	34.6	111.1	55.1	19.8	35.3	5.6	35.9	117.4	56.6																																																																																																																																																																																																			
HA-IM	<u>19.8</u>	<u>36.2</u>	5.3	36.2	115.1	55.3	20.3	36.2	5.8	36.2	124.1	56.8																																																																																																																																																																																																			
HA-SG	19.6	35.9	5.4	37.0	114.4	56.3	<b>20.9</b>	<b>37.1</b>	<u>6.0</u>	<b>38.1</b>	<b>129.8</b>	<b>57.6</b>																																																																																																																																																																																																			
+ GAT	19.4	35.8	<b>5.7</b>	36.2	112.7	56.0	19.7	35.3	5.9	36.2	123.6	<u>57.1</u>																																																																																																																																																																																																			
+ C-GAT	19.5	36.0	5.1	<u>37.6</u>	<u>116.4</u>	<u>56.1</u>	20.8	36.6	<b>6.0</b>	<u>37.2</u>	<u>127.3</u>	56.9																																																																																																																																																																																																			
	SPICE			Captioning			SPICE			Captioning																																																																																																																																																																																																					
	All	Obj	Rel	B4	C	R	All	Obj	Rel	B4	C	R																																																																																																																																																																																																			
BUTD	<b>20.5</b>	<b>37.0</b>	<u>5.5</u>	<b>38.4</b>	<b>117.5</b>	<b>57.1</b>	<b>20.9</b>	<u>36.8</u>	5.1	<u>37.2</u>	126.5	57.0																																																																																																																																																																																																			
FA	18.8	35.0	5.3	34.6	111.1	55.1	19.8	35.3	5.6	35.9	117.4	56.6																																																																																																																																																																																																			
HA-IM	<u>19.8</u>	<u>36.2</u>	5.3	36.2	115.1	55.3	20.3	36.2	5.8	36.2	124.1	56.8																																																																																																																																																																																																			
HA-SG	19.6	35.9	5.4	37.0	114.4	56.3	<b>20.9</b>	<b>37.1</b>	<u>6.0</u>	<b>38.1</b>	<b>129.8</b>	<b>57.6</b>																																																																																																																																																																																																			
+ GAT	19.4	35.8	<b>5.7</b>	36.2	112.7	56.0	19.7	35.3	5.9	36.2	123.6	<u>57.1</u>																																																																																																																																																																																																			
+ C-GAT	19.5	36.0	5.1	<u>37.6</u>	<u>116.4</u>	<u>56.1</u>	20.8	36.6	<b>6.0</b>	<u>37.2</u>	<u>127.3</u>	56.9																																																																																																																																																																																																			

(c) Average VG-COCO dataset	(d) High VG-COCO dataset
-----------------------------	--------------------------

Table 3: SPICE breakdown and captioning metrics for images in VG-COCO validation split. Results for the full VG-COCO, and for subsets of images collected according to the quality of their corresponding predicted scene graphs: low, average, and high. See Sections 5 for details on all models and acronyms. Metrics reported are: overall SPICE F1 score (All), object (Obj) and relation (Rel) F1 score components, BLEU-4 (B4), CIDEr (C), and ROUGE-L (R). We bold-face the best and underline the second-best overall scores per metric and per data subset.

the baseline and our best models according to all metrics. All these gains are very promising and suggest that when we have high quality scene graphs, we can expect a consistent positive transfer into image captioning models. However, the overall SPICE score is the highest for both BUTD and HA-SG, while BUTD has lower scores for objects and relations F-measure. That suggests that other components part of SPICE were worsened with the addition of scene graphs. Since this is not the focus of this paper, we did not investigate this further and leave that for future work.

Overall, these results show that indiscriminately using scene graphs from pretrained SGG models downstream on image captioning can be harmful because of the amount of noise present in these scene graphs. However, when this noise is smaller and the scene graphs of higher quality, our findings together suggest that scene graphs can be useful in image captioning models.

**Ground-truth graphs** Finally, we also conduct a small-scale experiment using ground-truth scene

graphs and evaluate how using these instead of predicted scene graphs at inference time impacts models, which can be found in Appendix B.

**Qualitative Results** Here, we try to determine if there is a clear difference in the difficulty in captioning images in *low*, *average*, and *high* quality sets, which might help explain the result in Table 3. In Figures 4 and 5 we show some images for the low and high scoring graphs, respectively. At a first glance, images from both sets appear equally cluttered with objects (i.e., which we hypothesise should correlate with the image being harder to describe). Furthermore, for both low and high scoring scene graphs, the average number of objects and relations is 23 and 22 respectively. However, we note that even scene graphs in the *high* quality set often include tiny objects and details, e.g. the image in the right of Figure 4 shows a single aircraft, but there are 17 annotated objects describing components such as wings, windows, etc.



Figure 4: Images, captions and ground-truth number of objects and relations for **high** scoring scene graph.



Figure 5: Images, captions and ground-truth number of objects and relations for **low** scoring scene graph.

## 6 Conclusions and Future Work

In this work, we investigate the impact scene graphs have on image captioning. We introduced conditional graph attention (C-GAT) networks and applied it to image captioning, and report promising results (Table 1). Overall, we found that improvements in captioning when using scene graphs generated with publicly available SGG models are minor. We observe a very small increase in the ability to describe relations as measured by relation SPICE F-scores, however, this is associated with models producing worse overall descriptions and producing lower object SPICE F-scores.

In an in-depth analysis, we found that the predicted scene graphs contain a large amount of noise which harms the captioning process. When this noise is reduced, large gains can be achieved across all image captioning metrics, e.g. 3.3 CIDEr points in the high VG-COCO split (Table 3d). This indicates that with further research and improved scene graph generation models, we will likely be able to observe consistent gains in image captioning and possibly other tasks by leveraging silver-standard scene graphs.

**Future work** In further research, we will conduct an in-depth analysis of our proposed condi-

tional graph attention to determine what tasks other than image captioning we can apply it to. We envision using it for visual question-answering also with generated scene graphs, and on syntax-aware neural machine translation (Bastings et al., 2017), fake news detection (Monti et al., 2019), and question answering (Zhang et al., 2018).

In a focused qualitative analysis, we found that the scene graphs represent objects and relations in images sometimes with great detail. We plan to investigate how to account for such highly detailed objects/relations in the context of image captioning. Finally, we will look into a method to use predicted scene graphs selectively according to their estimated quality, possibly selecting the best graph between those generated by different SGG models.

## Acknowledgments

We would like to thank the COST Action CA18231 for funding a research visit to collaborate on this project. This work is funded by the European Research Council (ERC) under the ERC Advanced Grant 788506. IC has received funding from the European Union’s Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No 838188.

## References

- Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2016. Spice: Semantic propositional image caption evaluation. In *Computer Vision – ECCV 2016*, pages 382–398, Cham. Springer International Publishing.
- Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.
- Joost Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Sima'an. 2017. Graph convolutional encoders for syntax-aware neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1957–1967, Copenhagen, Denmark. Association for Computational Linguistics.
- Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. 2018. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*.
- Ross Girshick. 2015. Fast R-CNN. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV ’15, page 1440–1448, USA. IEEE Computer Society.
- Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR ’14*, page 580–587, USA. IEEE Computer Society.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778.
- Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.
- Justin Johnson, Agrim Gupta, and Li Fei-Fei. 2018. Image generation from scene graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1219–1228.
- Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C. Lawrence Zitnick, and Ross Girshick. 2017. CLEVR: A Diagnostic Dataset for Compositional Language and Elementary Visual Reasoning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Justin Johnson, Ranjay Krishna, Michael Stark, Li-Jia Li, David Shamma, Michael Bernstein, and Li Fei-Fei. 2015. Image retrieval using scene graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3668–3678.
- Andrej Karpathy and Li Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137.
- Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Boris Knyazev, Harm de Vries, Cătălina Cangea, Graham W Taylor, Aaron Courville, and Eugene Belilovsky. 2020. Graph density-aware losses for novel compositions in scene graph generation. *arXiv preprint arXiv:2005.08230*.
- Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, and et al. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision*, 123(1):32–73.
- Guang Li, Linchao Zhu, Ping Liu, and Yi Yang. 2019. Entangled transformer for image captioning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.
- X. Li and S. Jiang. 2019. Know more say less: Image captioning based on scene graphs. *IEEE Transactions on Multimedia*, 21(8):2117–2130.
- Yikang Li, Wanli Ouyang, Bolei Zhou, Jianping Shi, Chao Zhang, and Xiaogang Wang. 2018. Factorizable net: An efficient subgraph-based framework for scene graph generation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 335–351.
- Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.
- Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.
- Jiasen Lu, Caiming Xiong, Devi Parikh, and Richard Socher. 2017. Knowing when to look: Adaptive attention via a visual sentinel for image captioning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

- Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. 2018. Neural baby talk. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7219–7228.
- Diego Marcheggiani, Anton Frolov, and Ivan Titov. 2017. A simple and accurate syntax-agnostic neural model for dependency-based semantic role labeling. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 411–420, Vancouver, Canada. Association for Computational Linguistics.
- Federico Monti, Fabrizio Frasca, Davide Eynard, Damon Mannion, and Michael M Bronstein. 2019. Fake news detection on social media using geometric deep learning. *arXiv preprint arXiv:1902.06673*.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, ACL '02*, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99.
- Steven J. Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. 2017. Self-critical sequence training for image captioning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.
- Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations*.
- Dalin Wang, Daniel Beck, and Trevor Cohn. 2019. On the role of scene graphs in image captioning. In *Proceedings of the Beyond Vision and LANguage: inTEGRating Real-world kNowledge (LANTERN)*, pages 29–34, Hong Kong, China. Association for Computational Linguistics.
- Danfei Xu, Yuke Zhu, Christopher B Choy, and Li Fei-Fei. 2017. Scene graph generation by iterative message passing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5410–5419.
- Jianwei Yang, Jiasen Lu, Stefan Lee, Dhruv Batra, and Devi Parikh. 2018. Graph R-CNN for scene graph generation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 670–685.
- X. Yang, K. Tang, H. Zhang, and J. Cai. 2019. Auto-encoding scene graphs for image captioning. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10677–10686.
- J. Yu, J. Li, Z. Yu, and Q. Huang. 2019a. Multi-modal transformer with multi-view visual representation for image captioning. *IEEE Transactions on Circuits and Systems for Video Technology*, pages 1–1.
- J. Yu, J. Li, Z. Yu, and Q. Huang. 2019b. Multi-modal transformer with multi-view visual representation for image captioning. *IEEE Transactions on Circuits and Systems for Video Technology*, pages 1–1.
- Rowan Zellers, Mark Yatskar, Sam Thomson, and Yejin Choi. 2018. Neural motifs: Scene graph parsing with global context. In *Conference on Computer Vision and Pattern Recognition*.
- Yuyu Zhang, Hanjun Dai, Zornitsa Kozareva, Alexander J Smola, and Le Song. 2018. Variational reasoning for question answering with knowledge graph. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

## A Implementation Details

All our models are trained until convergence using early stopping with a patience of 20 epochs and a maximum of 50 epochs. We use the Adamax optimizer (Kingma and Ba, 2014) with an initial learning rate of 0.002, which we decay with a factor of 0.8 after 8 epochs without improvements on the validation set. Dropout regularisation with a probability of 50% is applied on word embeddings and on the hidden state of the second LSTM layer  $h_2^{(t)}$  before it is projected to compute the next word probabilities. We use a beam size of 5 during evaluation. All hidden layers and embedding sizes are set to 1024. Models are all trained on a single 12GB NVIDIA GPU.

We use a fixed number of  $n = 36$  objects extracted with our pretrained Faster R-CNN. The number of objects and relations extracted with the pretrained Iterative Message Passing model varies according to the input image, i.e. a maximum of  $o = 100$  objects and of  $r = 2500$  relations.

## B Using Ground-Truth Graphs

	SPICE			Captioning		
	All	Obj	Rel	B4	C	R
FA	18.3	34.3	<b>5.3</b>	30.5	95.8	53.1
HA-IM	<b>19.0</b>	<b>35.2</b>	4.8	<u>33.5</u>	<b>106.0</b>	<b>54.8</b>
HA-SG	18.8	34.7	4.9	32.9	<u>104.5</u>	54.3
+ GAT	18.6	34.7	<u>5.2</u>	32.5	100.9	53.9
+ C-GAT	18.9	<u>34.8</u>	5.1	<b>33.6</b>	104.3	<u>54.5</u>

Table 4: Results for the full VG-COCO validation set using features extracted for ground-truth scene graphs. Models and acronyms are described in Sections 5. Metrics reported are: the overall SPICE F1 score (All) and its object (Obj) and relation (Rel) F1 score components, BLEU-4 (B4), CIDEr (C), and ROUGE-L (R). We bold-face the best and underline the second-best overall scores per metric.

In this small-scale experiment, we generate features for ground-truth scene graphs to determine if more a positive transfer can be achieved on image captioning models. For the VG-COCO dataset, we take all the ground-truth object and relation boxes and pass these through the pretrained Iterative Message Passing (IMP) model, instead of the RPN and RelPN proposed boxes. This is the same pretrained (IMP) model used in the other experiments.

Wang et al. (2019) also did a similar experiment,

however, they also trained their models using features from gold-standard scene graphs, whereas we only use them to evaluate models previously trained on predicted scene graph features. In Table 4 we show that when using ground-truth scene graphs results are worse than those obtained using predicted ones (Table 3a). One obvious explanation is the mismatch between training and testing data, with regards to quality and number of features. Models are trained on the predicted scene graphs, which have an average of 34 object and 48 relation features per image (probably noisy, as seen in Section 5.4), whereas ground-truth graphs have an average of 21 objects and 18 relations per image.

# Systematically Exploring Redundancy Reduction in Summarizing Long Documents

Wen Xiao and Giuseppe Carenini

Department of Computer Science

University of British Columbia

Vancouver, BC, Canada, V6T 1Z4

{xiaowen3, carenini}@cs.ubc.ca

## Abstract

Our analysis of large summarization datasets indicates that redundancy is a very serious problem when summarizing long documents. Yet, redundancy reduction has not been thoroughly investigated in neural summarization. In this work, we systematically explore and compare different ways to deal with redundancy when summarizing long documents. Specifically, we organize the existing methods into categories based on when and how the redundancy is considered. Then, in the context of these categories, we propose three additional methods balancing non-redundancy and importance in a general and flexible way. In a series of experiments, we show that our proposed methods achieve the state-of-the-art with respect to ROUGE scores on two scientific paper datasets, Pubmed and arXiv, while reducing redundancy significantly.<sup>1</sup>

## 1 Introduction

Summarization is the task of shortening a given document(s) while maintaining the most important information. In general, a good summarizer should generate a summary that is syntactically accurate, semantically correct, coherent, and non-redundant (Saggion and Poibeau, 2013). While extractive methods tend to have better performance on the first two aspects, they are typically less coherent and *more redundant* than abstractive ones, where new sentences are often generated by sentence fusion and compression, which helps detecting and removing redundancy (Lebanoff et al., 2019). Although eliminating redundancy has been initially and more intensely studied in the field of multi-document summarization (Lloret and Sanz, 2013), because important sentences selected from multiple documents (about the same topic) are more

likely to be redundant than sentences from the same document, generating a non-redundant summary should still be one of the goals for single document summarization (Lin et al., 2009).

Generally speaking, there is a trade-off between importance and diversity (non-redundancy) (Jung et al., 2019), which is reflected in the two phases, *sentence scoring* and *sentence selection* (Zhou et al., 2018) in which extractive summarization task can be naturally decomposed. The former typically scores sentences based on importance, while the latter selects sentences based on their scores, but also possibly taking other factors (including redundancy) into account.

Traditionally, in non-neural approaches the trade-off between importance and redundancy has been carefully considered, with sentence selection picking sentences by optimizing an objective function that balances the two aspects (Carbonell and Goldstein, 1998; Ren et al., 2016). In contrast, more recent works on neural extractive summarization models has so far over-emphasized sentence importance and the corresponding scoring phase, while paying little attention to how to reduce redundancy in the selection phase, where they simply apply a greedy algorithm to select sentences (e.g., Cheng and Lapata (2016); Xiao and Carenini (2019)). Notice that this is especially problematic for long documents, where redundancy tends to be a more serious problem, as we have observed in key datasets. Improving redundancy reduction in neural extractive summarization for long documents is a major goal of this paper.

Indeed, some recently proposed neural methods aim to reduce redundancy, but they either do that implicitly or inflexibly and only focusing on short documents (e.g., news). For instance, some models learn to reduce redundancy when predicting the scores (Nallapati et al., 2016a), or jointly learn to score and select sentences (Zhou et al., 2018) in

<sup>1</sup>Our code can be found here - <http://www.cs.ubc.ca/cs-research/lci/research-groups/natural-language-processing>

an implicit way. However, whether these strategies actually help reducing redundancy is still an open empirical question. The only neural attempt of explicitly reduce redundancy in the sentence selection phase is the Trigram Blocking technique, used in recent extractive summarization models on news datasets (e.g., (Liu and Lapata, 2019)). However, the effectiveness of such strategy on the summarization of long documents has not been tested. Finally, a very recent work by Bi et al. (2020) attempts to reduce redundancy in more sophisticated ways, but still focusing on news. Furthermore, since it relies on BERT, such model is unsuitable to deal with long documents (with over 3,000 words).

To address this rather confusing situation, characterized by unclear connections between all the proposed neural models, by their limited focus on short documents, and by spotty evaluations, in this paper we systematically organize existing redundancy reduction methods into three categories, and compare them with respect to the informativeness and redundancy of the generated summary for long documents. In particular, to perform a fair comparison we re-implement all methods by modifying a common basic model (Xiao and Carenini, 2019), which is a top performer on long documents without considering redundancy. Additionally, we propose three new methods that we argue will reduce redundancy more explicitly and flexibly in the sentence scoring and sentence selection phase by deploying more suitable decoders, loss functions and/or sentence selection algorithms, again building for a fair comparison on the common basic model (Xiao and Carenini, 2019).

To summarize, our main contributions in this paper are: we first examine popular datasets, and show that redundancy is a more serious problem when summarizing long documents (e.g., scientific papers) than short ones (e.g. news). Secondly, we not only reorganize and re-implement existing neural methods for redundancy reduction, but we also propose three new general and flexible methods. Finally, in a series of experiments, we compare existing and proposed methods on long documents (i.e., the Pubmed and arXiv datasets), with respect to ROUGE scores (Lin, 2004) and redundancy scores (Peyrard et al., 2017; Feigenblat et al., 2017).

As a preview, empirical results reveal that the proposed methods achieve state-of-the-art performance on ROUGE scores, on the two scientific paper datasets, while also reducing the redundancy

significantly.

## 2 Related Work

In traditional extractive summarization, the process is treated as a discrete optimization problem balancing between importance scores and redundancy scores, with techniques like Maximal Marginal Relevance(MMR)(Carbonell and Goldstein, 1998), redundancy-aware feature-based sentence classifiers (Ren et al., 2016) and graph-based submodular selection (Lin et al., 2009).

In recent years, researchers have explored neural extractive summarization solutions, which score sentences by training the neural models on a large corpus, and simply apply a greedy algorithm for sentence selection (Cheng and Lapata, 2016; Nallapati et al., 2016a). Although a model with a sequence decoder might plausibly encode redundancy information implicitly, Kedzie et al. (2018) empirically show that this is not the case, since non auto-regressive models (the ones scoring each sentence independently), perform on par with models with a sequence decoder. In one of our new methods, to effectively capture redundancy information, we specify a new loss that explicitly consider redundancy when training the neural model.

Beyond a greedy algorithm, the Trigram Blocking is frequently used to explicitly reduce redundancy in the sentence selection phase (Liu and Lapata, 2019). In essence, a new sentence is not added to the summary if it shares a 3-gram with the previously added one. Paulus et al. (2017) first adopt the strategy for abstractive summarization, which forces the model not to produce the same trigram twice in the generated summaries, as a simplified version of MMR (Carbonell and Goldstein, 1998). Arguably, this method is too crude for documents with relatively long sentences or specific concentrations (e.g. scientific papers), where some technical terms, possibly longer than 2-grams, are repeated frequently in the ‘important sentences’ (even in the reference summaries). To address this limitation, we propose a neural version of MMR to deal with redundancy within the sentence selection phase in a more flexible way, that can be tuned to balance importance and non-redundancy as needed.

The idea of MMR has also inspired Zhou et al. (2018), who propose a model jointly learning to score and select the sentences. Yet, this work not only focuses on summarizing short documents (i.e., news), but also uses MMR implicitly, and arguably

sub-optimally, by learning a score that only indirectly captures the trade-off between relevance and redundancy. To improve on this approach, in this paper we propose a third new method, in which importance and redundancy are explicitly weighted, while still making the sentence scoring and selection benefit from each other by fine tuning the trained neural model through a Reinforcement Learning (RL) mechanism.

Finally, Bi et al. (2020) is the most recent (still unpublished) work on reducing redundancy in neural single document summarization. However, their goal is very different from ours, since they focus on relatively short documents in the news domain.

### 3 Measuring Redundancy: metrics and comparing long vs. short documents

We use the following two relatively new metrics to measure redundancy in the source documents and in the generated summaries.

**Unique n-gram ratio<sup>2</sup>:** proposed in Peyrard et al. (2017), it measures n-grams uniqueness; the lower it is, the more redundant the document is.

$$\text{Uniq\_ngram\_ratio} = \frac{\text{count}(\text{uniqu\_n\_gram})}{\text{count}(n\_gram)}$$

**Normalized Inverse of Diversity (NID):** captures redundancy, as the inverse of a diversity metric with length normalization. Diversity is defined as the entropy of unigrams in the document (Feigenblat et al., 2017). Since longer documents are more likely to have a higher entropy, we normalize the diversity with the maximum possible entropy for the document  $\log(|D|)$ . Thus, we have:

$$NID = 1 - \frac{\text{entropy}(D)}{\log(|D|)}$$

Note that higher NID indicates more redundancy.

When we compare the redundancy of long vs. short documents with respect to these two metrics on four popular datasets for summarization (CNNDM (Nallapati et al., 2016b), Xsum (Narayan et al., 2018), Pubmed and arXiv (Cohan et al., 2018)), we observe that long documents are substantially more redundant than short ones (as it was already pointed out in the past (Stewart and Carbonell, 1998)). Table 1 shows the basic statistics of each dataset, along with the average NID

<sup>2</sup>In this paper, all the unique n-gram ratios are shown in percentage.

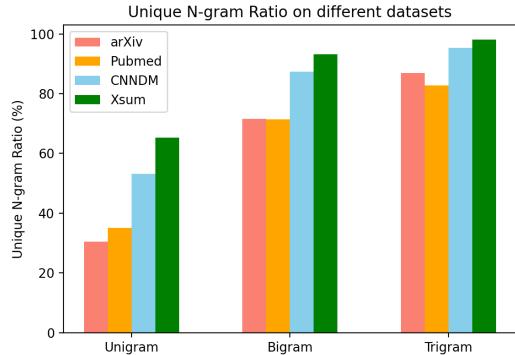


Figure 1: The average unique n-gram ratio in the documents across different datasets. To reduce the effect of length difference, stopwords were removed.

scores, while Figure 1 shows the average Unique n-gram Ratio for the same datasets. These observations provide further evidence that redundancy is a more serious problem in long documents. In addition, notice that the sentences in the scientific paper datasets are much longer than in the news datasets, which plausibly makes it even harder to balance between importance and non-redundancy.

Datasets	# Doc.	# words/doc.	# words/sent.	NID
Xsum	203k	429	22.8	0.188
CNNDM	270k	823	19.9	0.205
Pubmed	115k	3142	35.1	0.255
arXiv	201k	6081	29.2	0.267

Table 1: Longer documents are more redundant

### 4 Redundancy Reduction Methods

We systematically organize neural redundancy reduction methods into three categories, and compare prototypical methods from each category.

- A The decoder is designed to implicitly take redundancy into account.
- B In the sentence scoring phase, explicitly learn to reduce the redundancy.
- C In the sentence selection phase, select sentences with less redundancy.

In this section, we describe different methods from each category. To compare them in a fair way, we build all of them on a basic ExtSum-LG model (see §4.1), by modifying the decoder and the loss function in the sentence selection phase or the sentence selection algorithm. In Table 2, we summarize the architecture (Encoder, Decoder, Loss Function and sentence selection algorithm) of all the methods we compare.

Categ.	Methods	Sent. Scor.			Sent. Sel.
		Encoder	Decoder	Loss Func.	
-	Naive MMR		Cosine Similarity		MMR Select
-	ExtSum-LG	Encoder-LG	MLP	Cross Entropy (CE)	Greedy
A	+ SR Decoder	Encoder-LG	SR Decoder	CE	Greedy
A	+ NeuSum Decoder	Encoder-LG	NeuSum Decoder	KL Divergence	NeuSum Decoder
B	<b>+ RdLoss</b>	Encoder-LG	MLP	CE + Red. Loss1	Greedy
C	+ Trigram Blocking	Encoder-LG	MLP	CE	Trigram Blocking
C	<b>+ MMR-Select</b>	Encoder-LG	MLP	CE	MMR Select
C	<b>+ MMR-Select+</b>	Encoder-LG	MLP	CE + Red. Loss2	MMR Select

Table 2: The architecture of redundancy reduction methods. **Bold** methods are proposed in this paper.

#### 4.1 Baseline Models

We consider two baseline models. One is an influential unsupervised method explicitly balancing importance and redundancy (Naive MMR). The other is our basic neural supervised model not dealing with redundancy at all (ExtSum-LG), to which we add different redundancy reduction mechanisms.

##### Naive MMR

MMR (Carbonell and Goldstein, 1998) is a traditional extractive summarization method, which re-ranks the candidate sentences with a balance between query-relevance(importance) and information novelty(non-redundancy). Given a document  $D$ , at each step, MMR selects one sentence from the candidate set  $D \setminus \hat{S}$  that is relevant with the query  $Q$ , while containing little redundancy with the current summary  $\hat{S}$ . Note that if there is no specific query, then the query is the representation of the whole document. The method can be formally specified as:

$$\begin{aligned} MMR = \arg \max_{s_i \in D \setminus \hat{S}} & [\lambda Sim_1(s_i, Q) \\ & - (1 - \lambda) \max_{s_j \in \hat{S}} Sim_2(s_i, s_j)] \end{aligned}$$

where  $Sim_1(s_i, Q)$  measures the similarity between the candidate sentence  $s_i$  and the query, indicating the importance of  $s_i$ , while  $\max_{s_j \in \hat{S}} Sim_2(s_i, s_j)$  measures the similarity between the candidate sentence  $s_i$  and the current summary  $\hat{S}$ , representing the redundancy, and  $\lambda$  is the balancing factor. In this work, all the  $Sim$  are computed as the cosine similarity between the embeddings of the sentences.

##### ExtSum-LG

For the basic model, we use the current state-of-the-art model (Xiao and Carenini, 2019) on the summarization of long documents. It is a novel extractive summarization model incorporating local context and global context in the encoder, with

an MLP layer as decoder and cross-entropy as the loss function. For the sentence selection phase, it greedily picks the sentences according to the score predicted by the neural model. In this method, redundancy is not considered, so it is a good testbed for adding and comparing redundancy reduction methods.

Specifically, for a document  $D = \{s_1, s_2, \dots, s_n\}$ , the output of the encoder is  $h_i$  for each sentence  $s_i$ , and the decoder gives output  $P(y_i)$  as the confidence score on the importance of sentence  $s_i$ . Finally, the model is trained on the Cross Entropy Loss :

$$L_{ce} = - \sum_{i=1}^n (y_i \log P(y_i) + (1 - y_i) \log (1 - P(y_i)))$$

#### 4.2 Implicitly Reduce Redundancy in the neural model (Category A, Table 2)

In this section, we describe two decoders from previous work, in which the redundancy of the summary is considered implicitly.

**SummaRuNNer Decoder:** Nallapati et al. (2016a) introduce a decoder that computes a sentence score based on its salience, novelty(non-redundancy) and position to decide whether it should be included in the summary. Formally:

$$\begin{aligned} P(y_i) = \sigma(W_c h_i) & \quad \#Content \\ & + h_i W_s d \quad \#Salience \\ & - h_i^T W_r \tanh(summ_i) \quad \#Novelty \\ & + W_{ap} p_i^a + W_{rp} p_i^r \quad \#Position \\ & + b) \quad \#Bias \end{aligned}$$

where  $h_i$  is the hidden state of sentence  $i$  from the encoder,  $d$  is the document representation,  $summ_i$  is the summary representation, updated after each decoding step, and  $p_i^a, p_i^r$  are absolute and relative position embeddings, respectively. Once  $P(y_i)$  is obtained for each sentence  $i$ , a greedy algorithm selects the sentences to form the final summary.

Notice that although SummaRuNNer does contain a component assessing novelty, it would be inappropriate to view this model as explicitly dealing with redundancy because the novelty component is not directly supervised.

**NeuSum Decoder:** One of the main drawback of SummaRuNNer decoder is that it always score the sentences in order, i.e., the former sentences are not influenced by the latter ones. In addition, it only considers redundancy in the sentence scoring phase, while simply using a greedy algorithm to select sentences according to the resulting scores. To address these problems, Zhou et al. (2018) propose a new decoder to identify the relative gain of sentences, jointly learning to score and select sentences. In such decoder, instead of feeding the sentences and getting the scores in order, they use a mechanism similar to the pointer network (Vinyals et al., 2015) to predict the scores of all the sentences at each step, select the sentence with the highest score, and feed it to the next step of sentence selection. As for the loss function, they use the KL divergence between the predicted score distribution and the relative ROUGE F1 gain at each step. To be specific, the loss computed at step  $t$  is:

$$\begin{aligned} L_t &= D_{KL}(P_t || Q_t) \\ P_t(y_i) &= \frac{\exp(\sigma(h_i))}{\sum_{j=1}^n \exp(\sigma(h_j))} \\ Q_t(y_i) &= \frac{\exp(\tau \tilde{g}_t(y_i))}{\sum_{j=1}^n \exp(\tau \tilde{g}_t(y_j))} \\ g_t(y_i) &= r1(\mathbb{S}_{t-1} \cup s_i) - r1(\mathbb{S}_{t-1}) \end{aligned}$$

where  $P_t$ ,  $Q_t$  are the predicted and ground truth relative gain respectively,  $g_t(y_i)$  is the ROUGE F1 gain with respect to the current partial summary  $\mathbb{S}_{t-1}$  for sentence  $s_i$ , and  $\tilde{g}_t(y_i)$  is the Min-Max normalized  $g_t(y_i)$ .  $\tau$  is a smoothing factor, which is set to 200 empirically on the Pubmed dataset.<sup>3</sup>

### 4.3 Explicitly Reduce Redundancy in Sentence Scoring (Category B, Table 2)

We propose a new method to explicitly learn to reduce redundancy when scoring the sentences.

**RdLoss:** Although Zhou et al. (2018) jointly train the decoder to score and select sentences, it still learns to reduce redundancy implicitly, and the method does not allow controlling the degree of redundancy. To address this limitation, we propose a rather simple method to explicitly force the model

<sup>3</sup>Due to the complexity of generating the target distribution  $Q$ , we only experiment with this method on Pubmed.

to reduce redundancy in the sentence scoring phase by adding a redundancy loss term to the original loss function, motivated by the success of a similar strategy of adding a bias loss term in the gender debiasing task (Qian et al., 2019). Our new loss term  $L_{rd}$  is naturally defined as the expected redundancy contained in the resulting summary, as shown below:

$$\begin{aligned} L &= \beta L_{ce} + (1 - \beta)L_{rd} \\ L_{rd} &= \sum_{i=1}^n \sum_{j=1}^n P(y_i)P(y_j)Sim(s_i, s_j) \end{aligned}$$

where  $P(y_i)$ ,  $P(y_j)$  are the confidence scores of sentence  $i$  and  $j$  on whether to select the sentences in the generated summary, and  $Sim(s_i, s_j)$  is the similarity, i.e. redundancy between sentence  $i$  and  $j$ .<sup>4</sup> By adding the redundancy loss term, we penalize it more if two sentences are similar to each other and both of them have high confidence scores.  $\beta$  is a balance factor, controlling the degree of redundancy.

### 4.4 Explicitly Reduce Redundancy in Sentence Selection (Category C, Table 2)

We first introduce an existing method and then propose two novel methods that explicitly reduce redundancy in the sentence selection phase.

**Trigram Blocking** is widely used in recent extractive summarization models on the news dataset (e.g. Liu and Lapata (2019)). Intuitively, it borrows the idea of MMR to balance the importance and non-redundancy when selecting sentences. In particular, given the predicted sentence scores, instead of just selecting sentences greedily according to the scores, the current candidate is added to the summary only if it does not have trigram overlap with the previous selected sentences. Otherwise, the current candidate sentence is ignored and the next one is checked, until the length limit is reached.

**MMR-Select:** Inspired by the existence of a relevance/redundancy trade-off, we propose MMR-Select, a simple method to eliminate redundancy when a neural summarizer selects sentences to form a summary, in a way that is arguably more flexible than Trigram Blocking with a balance factor  $\lambda$ .

With the confidence score computed by the basic model,  $P = \{P(y_1), P(y_2), \dots, P(y_n)\}$ , instead of picking sentences greedily, we pick the sentences according to the MMR-score, which is defined

<sup>4</sup>Noting that we define  $Sim(s_i, s_i)$  as 0

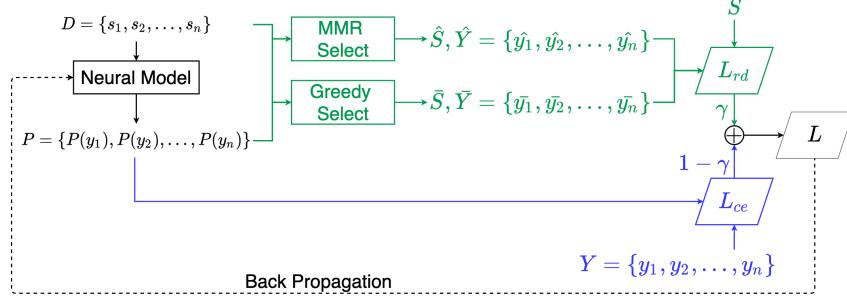


Figure 2: The pipeline of the MMR-Select+ method, where  $\hat{S}$ ,  $\hat{Y}$  and  $\bar{S}$ ,  $\bar{Y}$  are the summary and labels generated by the MMR-Select algorithm and the normal greedy algorithm, respectively.  $S$  and  $Y$  are the ground truth summary and the oracle labels.

based on MMR and updated after each single sentence being selected.

$$\text{MMR-Select} = \arg \max_{s_i \in D \setminus \hat{S}} [\text{MMR-score}_i]$$

$$\text{MMR-score}_i = \lambda P(y_i) - (1 - \lambda) \max_{s_j \in \hat{S}} \text{Sim}(s_i, s_j)$$

The main difference between the Naive MMR and MMR-Select falls into the computation of the importance score. In the Naive MMR, the importance score is the similarity between each sentence and the query, or the whole document, while in MMR-Select, the importance score is computed by a trained neural model.

**MMR-Select+ :** The main limitation of MMR-Select is that the sentence scoring phase and the sentence selection phase cannot benefit from each other, because they are totally separate.

To promote synergy between these two phases, we design a new method, MMR-Select+, shown in Figure 2, which synergistically combines three components: the basic model, the original cross-entropy loss  $L_{ce}$  (in blue), and an RL mechanism (in green) whose loss is  $L_{rd}$ . The neural model is then trained on a mixed objective loss  $L$  with  $\gamma$  as the scaling factor. Zooming on the details of the RL component, it first generates a summary  $\hat{S}$  by applying the MMR selection described for MMR-Select, which is to greedily pick sentences according to MMR-score, as well as the corresponding label assignment  $\hat{Y} = \{\hat{y}_1, \hat{y}_2, \dots, \hat{y}_n\}$  ( $\hat{y}_i = 1$  if  $s_i$  is selected,  $\hat{y}_i = 0$  otherwise). Then, the expected reward is computed based on the ROUGE score between  $\hat{S}$  and the gold-standard human abstractive summary  $S$  weighted by the probability of the  $\hat{Y}$  labels. Notice that we also adopt the self-critical strategy (Paulus et al., 2017) to help accelerating the convergence by adding a baseline summary  $\bar{S}$ , which is generated by greedily picking

the sentences according to  $P$ .  $r(\bar{S})$  is the reward of this baseline summary and it is subtracted from  $r(\hat{S})$  to only positively reward summaries which are better than the baseline. Formally, the whole MMR-Select+ model can be specified as follows:

$$L = \gamma L_{rd} + (1 - \gamma) L_{ce}$$

$$L_{rd} = -(r(\hat{S}) - r(\bar{S})) \sum_{i=1}^n \log P(\hat{y}_i)$$

$$r(S') = \frac{1}{3} \sum_{k \in \{1, 2, L\}} \text{ROUGE-k}(S', S)$$

## 5 Experiments

In this section, we describe the settings, results and analysis of the experiments of different methods on the Pubmed and arXiv datasets.

### 5.1 Model Settings

Following previous work, we use GloVe (Pennington et al., 2014) as word embedding, and the average word embedding as the distributed representation of sentences. To be comparable with Xiao and Carenini (2019), we set word length limit of the generated summaries as 200 on both datasets.

<sup>6</sup> We tune the hyperparameter  $\lambda$  and  $\beta$  in the respective methods on the validation set, and set  $\lambda = 0.6$ ,  $\beta = 0.3$  for both datasets. Following previous work (e.g., Li et al. (2019)),  $\gamma$  was set to 0.99. For training MMR-Select+, the learning rate is  $lr = 1e - 6$ ; we start with the pretrained ExtSumm-LG model. As for the evaluation metric, we use ROUGE scores as the measurement of importance while using the Unique N-gram Ratio and NID defined in Section 3 as the measurements of redundancy.

<sup>5</sup>The results of ExtSum-LG were obtained by re-running their model.

<sup>6</sup>A document representation in Unsupervised MMR is similarly computed by averaging the embeddings of all the words.

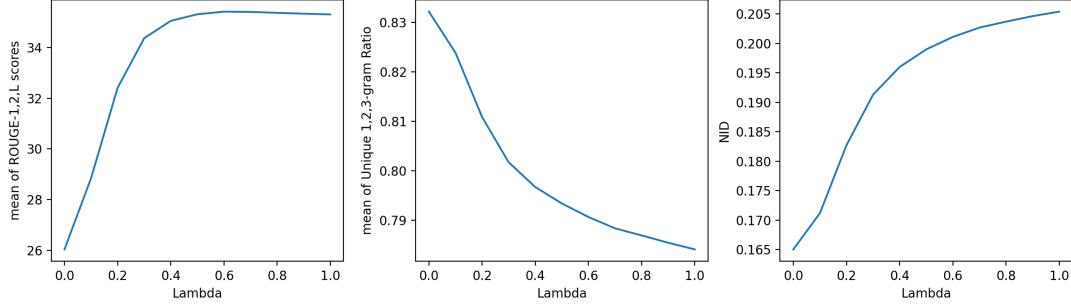


Figure 3: The average ROUGE scores, average unique n-gram ratios, and average NID scores with different  $\lambda$  used in the MMR-Select on the validation set. Remember that the higher the Unique n-gram Ratio, the lower NID, the less redundancy contained in the summary.

Categ.	Model	Pubmed			arXiv		
		ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-1	ROUGE-2	ROUGE-L
C	Naive MMR	37.46	11.25	32.22	33.74	8.50	28.36
-	ExtSum-LG <sup>5</sup>	45.18	20.20	40.72	43.77	17.50	38.71
A	+SR Decoder	45.18	20.16	40.69	43.92	17.65	38.83
A	+NeuSum Decoder	44.54	19.66	40.42	-	-	-
B	+RdLoss	45.30†	20.42†	40.95†	44.01†	17.79†	39.09†
C	+Trigram Blocking	43.33	17.67	39.01	42.75	15.73	37.85
C	+MMR-Select	45.29†	20.30†	40.90†	43.81	17.41	38.94
C	+MMR-Select+	45.39†	20.37†	40.99†	43.87†	17.50	38.97†
-	Oracle	55.05	27.48	49.11	53.89	23.07	46.54

Table 3: Rouge score of different summarization models on the Pubmed and arXiv datasets. † indicates significantly better than the ExtSum-LG with confidence level 99% on the Bootstrap Significance test. Green numbers means it's better than ExtSum-LG on the certain metric, and the red numbers means worse.

Categ.	Model	Pubmed				arXiv			
		Unigram%	Bigram%	Trigram%	NID	Unigram%	Bigram%	Trigram%	NID
C	Naive MMR	56.55	90.93	96.95	0.1881	53.01	88.82	96.28	0.1992
-	ExtSum-LG	53.02	87.29	94.37	0.2066	52.17	87.19	95.38	0.2088
A	+SR Decoder	52.88	87.17	94.32	0.2070	51.98	87.08	95.31	0.2097
A	+NeuSum Decoder	54.88†	88.71†	95.13†	0.1993†	-	-	-	-
B	+RdLoss	53.23†	87.41	94.43	0.2052†	52.17	87.20	95.36	0.2085
C	+Trigram Blocking	57.58†‡	93.05†‡	98.56†‡	0.1818†‡	56.12†‡	92.38†‡	98.94†‡	0.1876†‡
C	+MMR-Select	53.76†	88.04†	94.96†	0.2022	52.80†	87.64†	95.40	0.2055†
C	+MMR-Select+	53.93†	88.32	95.14	0.2014	52.76†	87.78†	95.70†	0.2055†
-	Oracle	56.66	89.25	95.55	0.2036	56.74	90.81	96.82	0.2029
-	Reference	56.69	89.45	95.95	0.2005	58.92	90.13	97.02	0.1970

Table 4: Unique n-gram ratio and NID score on the two datasets. † indicates significant differences from (Xiao and Carenini, 2019) with confidence level 99%, while ‡ indicates significant differences from all the other models with confidence level 99% on the Bootstrap Significance test. Noting the higher the Unique n-gram Ratio, the lower NID, the less redundancy contained in the summary. Green numbers means it's better than ExtSum-LG on the certain metric, and the red numbers means worse.

## 5.2 Finetuning $\lambda$

Consistently with previous work (Jung et al., 2019), when we finetune  $\lambda$  of MMR Select on the validation set, we pinpoint the trade off between importance and non-redundancy in the generated summary (see Figure 3). For  $\lambda \leq 0.6$ , as we increase the weight of importance score, the average ROUGE scores continuously increase while the redundancy/diversity increases/drops rapidly. But since extractive methods can only reuse sentences from the input document, there is an upper bound on how much the generated summary can match the ground-truth summary, so when

$\lambda > 0.6$ , the ROUGE score even drops by a small margin, while the redundancy/diversity still decreases/drops. Then the problem to solve for future work is how to increase the peak, which could be done by either applying finer units (e.g., clauses instead of sentences) or further improve the model that predicts the importance score.

## 5.3 Overall Results and Analysis

The experimental results for the ROUGE scores are shown in Table 3, whereas results for redundancy scores (Unique N-gram Ratio and NID score) are shown in Table 4. With respect to the balance be-

tween importance and non-redundancy, despite the trade-off between the two aspects, all of the three methods we propose can reduce redundancy significantly while also improving the ROUGE score significantly compared with the ExtSum-LG basic neural model. In contrast, the NeuSum Decoder and Trigram Blocking effectively reduce redundancy, but in doing that they hurt the importance aspect considerably. Even worse, the SR Decoder is dominated by the basic model on both aspects.

Focusing on the redundancy aspect (Table 4), Trigram Blocking makes the largest improvement on redundancy reduction, but with a large drop in ROUGE scores. This is in striking contrast with results on news datasets (Liu and Lapata, 2019), where Trigram Blocking reduced redundancy while also improving the ROUGE score significantly. Plausibly, the difference between the performances across datasets might be the result of the inflexibility of the method. In both Pubmed and arXiv datasets, the sentences are much longer than those in the news dataset (See Table 1), and therefore, simply dropping candidate sentences with 3-gram overlap may lead to incorrectly missing sentences with substantial important information.

Furthermore, another insight revealed in Table 4 is that dealing with redundancy in the sentence selection phase is consistently more effective than doing it in the sentence scoring phase, regardless of whether this happens implicitly (NeuSum > SR Decoder) or explicitly (Trigram Blocking, MMR-Select/+ > RdLoss).

Moving to more specific findings about particular systems, we already noted that while the NeuSum Decoder reduces redundancy effectively, it performs poorly on the ROUGE score, something that did not happen with news datasets. A possible explanation is that the number of sentences selected for the scientific paper datasets (on average 6-7 sentences) is almost twice the number of sentences selected for news; and as it was mentioned in the original paper (Zhou et al., 2018), the precision of NeuSum drops rapidly after selecting a few sentences.

Other results confirm established beliefs. The considerable difference between Naive MMR and MMR-Select was expected given the recognized power of neural network over unsupervised methods. Secondly, the unimpressive performance of the SR decoder confirms that the in-order sequence scoring is too limited for effectively predicting im-

portance score and reducing redundancy.

#### 5.4 More Insights of the Experiments

In addition to the main experiment results discussed above, we further explore the performance on informativeness (ROUGE score) and redundancy (Unique N-gram Ratio) of different redundancy reduction methods under two different conditions, namely the degree of redundancy and the length of the source documents. Figure 4 shows the results on the Pubmed dataset, while further results of a similar analysis on the arXiv dataset can be found in the Appendices. With respect to the degree of redundancy, (upper part of Figure 4), the less redundant the document is, the less impact the redundancy reduction methods have. Among all the methods, although Trigram Blocking works the best with respect to reducing redundancy, it hurts the informativeness the most. However, it is still a good choice for a rather less redundant document (e.g. the documents in the last two bins with avg Unique N-gram Ratio over 0.7), which is also consistent with the previous works showing the Trigram Blocking works well on the news datasets, which tends to be less redundant (see §3). As for all the other methods, although they have the same trends, MMR-Select+ performs the best on both informativeness and redundancy reduction, especially for the more redundant documents.

Regarding to the length of the source document (bottom part of Figure 4), as the document become longer, both informativeness and redundancy in the summary generated by all methods increases and then decrease once hitting the peak. MMR-Select+ and MMR-Select are the best choices to balance between the informativeness and redundancy - they are the only two methods having the higher ROUGE scores and higher Unique N-gram ratios across different lengths, even for the short documents with less than 50 sentences.

Besides, we also conduct experiments on generating summaries with different length limit, where we found that our new methods are stable across different summary lengths (Figure. 5).

## 6 Conclusion and Future work

Balancing sentence importance and redundancy is a key problem in extractive summarization. By examining large summarization datasets, we find that longer documents tend to be more redundant. Therefore in this paper, we systematically explore and compare existing and newly proposed methods

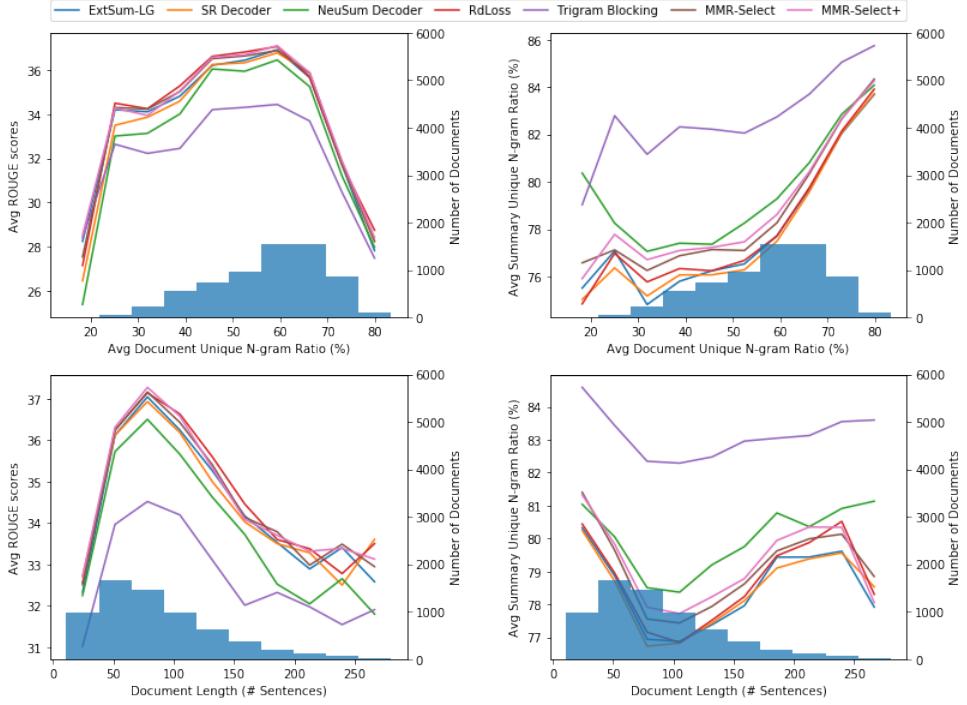


Figure 4: Comparing the average ROUGE scores and average unique n-gram ratios of different models on the Pubmed dataset, conditioned on different degrees of redundancy and lengths of the document (extremely long documents - i.e., 1% of the dataset are not shown because of space constraints).<sup>7</sup>

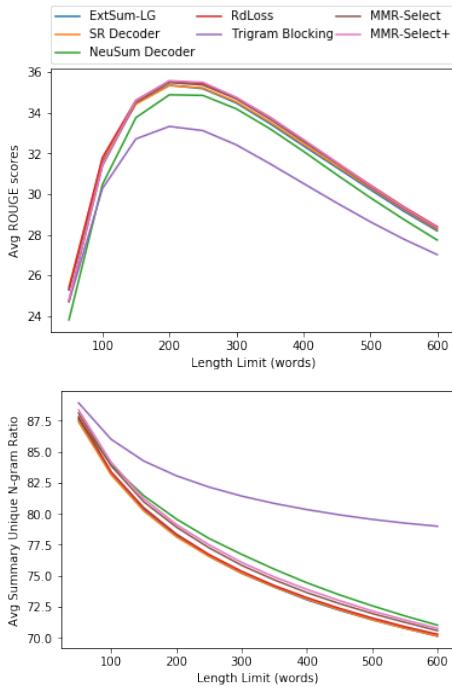


Figure 5: Comparing the average ROUGE scores and average unique n-gram ratios of different models with different word length limits on the Pubmed dataset. See Appendices for similar results on arXiv.

for redundancy reduction in summarizing long documents. Experiments indicate that our novel methods achieve SOTA on the ROUGE scores, while significantly reducing redundancy on two scientific paper datasets (Pubmed and arXiv). Interestingly, we show that redundancy reduction in sentence selection is more effective than in the sentence scoring phase, a finding to be further investigated .

Additional venues for future work include experimenting with generating summaries at finer granularity than sentences, as suggested by our analysis of the  $\lambda$  parameter. We also intend to explore other ways to assess redundancy, moving from computing the cosine similarity between sentence embeddings, to a pre-trained neural model for sentence similarity. Finally, we plan to run human evaluations to assess the quality of the generated summaries. This is quite challenging for scientific papers, as it requires participants to possess sophisticated domain-specific background knowledge.

## Acknowledgments

We thank reviewers and the UBC-NLP group for their insightful comments. This research was supported by the Language & Speech Innovation Lab of Cloud BU, Huawei Technologies Co., Ltd.

## References

- Keping Bi, Rahul Jha, W. Bruce Croft, and Asli Celikyilmaz. 2020. **ARED SUM: Adaptive Redundancy-Aware Iterative Sentence Ranking for Extractive Document Summarization.**
- Jaime Carbonell and Jade Goldstein. 1998. **The use of mmr, diversity-based reranking for reordering documents and producing summaries.** In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '98*, pages 335–336, New York, NY, USA. ACM.
- Jianpeng Cheng and Mirella Lapata. 2016. **Neural summarization by extracting sentences and words.** In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 484–494, Berlin, Germany. Association for Computational Linguistics.
- Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. **A discourse-aware attention model for abstractive summarization of long documents.** *CoRR*, abs/1804.05685.
- Guy Feigenblat, Haggai Roitman, Odellia Boni, and David Konopnicki. 2017. **Unsupervised query-focused multi-document summarization using the cross entropy method.** In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '17*, page 961–964, New York, NY, USA. Association for Computing Machinery.
- Taehee Jung, Dongyeop Kang, Lucas Mentch, and Edward Hovy. 2019. **Earlier Isn't Always Better: Sub-aspect Analysis on Corpus and System Biases in Summarization.** pages 3322–3333.
- Chris Kedzie, Kathleen McKeown, and Hal Daumé III. 2018. **Content selection in deep learning models of summarization.** In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1818–1828, Brussels, Belgium. Association for Computational Linguistics.
- Logan Lebanoff, John Muchovej, Franck Dernoncourt, Doo Soon Kim, Seokhwan Kim, Walter Chang, and Fei Liu. 2019. **Analyzing sentence fusion in abstractive summarization.** In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 104–110, Hong Kong, China. Association for Computational Linguistics.
- Siyao Li, Deren Lei, Pengda Qin, and William Yang Wang. 2019. **Deep reinforcement learning with distributional semantic rewards for abstractive summarization.** In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6040–6046, Hong Kong, China. Association for Computational Linguistics.
- Chin-Yew Lin. 2004. **ROUGE: A package for automatic evaluation of summaries.** In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.
- Hui Lin, Jeff Bilmes, and Shasha Xie. 2009. **Graph-based submodular selection for extractive summarization.** *Proceedings of the 2009 IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU 2009*, pages 381–386.
- Yang Liu and Mirella Lapata. 2019. **Text summarization with pretrained encoders.** In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740, Hong Kong, China. Association for Computational Linguistics.
- Elena Lloret and Manuel Sanz. 2013. **Tackling redundancy in text summarization through different levels of language analysis.** *Computer Standards Interfaces*, 35:507–518.
- Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2016a. **Summarunner: A recurrent neural network based sequence model for extractive summarization of documents.** *CoRR*, abs/1611.04230.
- Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gülcehre, and Bing Xiang. 2016b. **Abstractive text summarization using sequence-to-sequence RNNs and beyond.** In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany. Association for Computational Linguistics.
- Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. **Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization.** In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.
- Romain Paulus, Caiming Xiong, and Richard Socher. 2017. **A deep reinforced model for abstractive summarization.** *CoRR*, abs/1705.04304.
- Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. **Glove: Global vectors for word representation.** In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.
- Maxime Peyrard, Teresa Botschen, and Iryna Gurevych. 2017. **Learning to score system summaries for better content selection evaluation.** In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 74–84, Copenhagen, Denmark. Association for Computational Linguistics.
- Yusu Qian, Urwa Muaz, Ben Zhang, and Jae Won Hyun. 2019. **Reducing gender bias in word-level language models with a gender-equalizing loss function.** In *Proceedings of the 57th Annual Meeting of*

- the Association for Computational Linguistics: Student Research Workshop*, pages 223–228, Florence, Italy. Association for Computational Linguistics.
- Pengjie Ren, Furu Wei, Zhumin Chen, Jun Ma, and Ming Zhou. 2016. *A redundancy-aware sentence regression framework for extractive summarization*. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 33–43, Osaka, Japan. The COLING 2016 Organizing Committee.

Horacio Saggion and Thierry Poibeau. 2013. *Automatic Text Summarization: Past, Present and Future*, pages 3–21. Springer Berlin Heidelberg, Berlin, Heidelberg.

Jade Stewart and Jaime Carbonell. 1998. *Summarization: (1) using mmr for diversity - based reranking and (2) evaluating summaries*, pages 181–195.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. *Pointer networks*. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2692–2700. Curran Associates, Inc.

Wen Xiao and Giuseppe Carenini. 2019. *Extractive summarization of long documents by combining global and local context*. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3002–3012, Hong Kong, China. Association for Computational Linguistics.

Qingyu Zhou, Nan Yang, Furu Wei, Shaohan Huang, Ming Zhou, and Tiejun Zhao. 2018. *Neural document summarization by jointly learning to score and select sentences*. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 654–663, Melbourne, Australia. Association for Computational Linguistics.

## A Appendices

In these Appendices, we show more analysis of the experimental results.

### A.1 Analysis on arXiv Dataset under conditions

Figure 6 shows the performance on informativeness (ROUGE score) and redundancy (Unique N-gram Ratio) of different redundancy reduction methods under different conditions on the arXiv dataset. Comparing with the Pubmed dataset, the documents in the arXiv dataset tend to be longer and more redundant, as the majority of the documents in the Pubmed dataset have less than 100 sentences with average Unique N-gram Ratio in the 0.5 – 0.6

interval, while the majority of the documents in the arXiv dataset have number of sentences in the range 100 to 300 with average Unique N-gram Ratio in the 0.6 – 0.7 interval. Consistent with the result on the Pubmed dataset, the Trigram Blocking method is the best choice for rather less redundant documents (with average Unique N-gram Ratio larger than 0.7), and the MMR-Select+ is the one better or equivalent to the original model across different degree of redundancy, ignoring the outliers. With respect to the length of the documents, the MMR-Select+ and MMR-Select are consistently the most effective methods for balancing redundancy and informativeness on documents with different length.

### A.2 Analysis on Selection Overlap

To explore the difference made by applying different redundancy reduction methods on the original method(ExtSumLG), we compare the selected sentences by all the methods, and show the overlap ratios between every two methods, as well as the total number and the average length of selected sentences in the test set, in Table 5 and Table 6 for Pubmed dataset and arXiv dataset respectively. As we can see from the tables, except for the SR Decoder, all the other methods tend to select more and shorter sentences than the original summarizer. Regarding the overlap between the original method and the others, we observe that among all the three categories, the methods in category A tend to produce large differences, since these methods change the structure of the original model. Comparing the methods in Category C, around 36% of the sentences are regarded as redundant by Trigram Blocking, which means 36% of the sentences have trigram-overlap with other selected sentences, while only around 10% sentences are regarded as redundant by MMR-Select. As the ROUGE scores of MMR-Select are much better than Trigram Blocking on both datasets, this is in line with our analysis in Section 5.3, Trigram Blocking dropping some important sentences incorrectly. Interestingly, we notice that the overlap ratio between Trigram Block and MMR-Select is considerably larger than the overlap ratio of Trigram Block with original method (ExtSumLG) on both datasets. This indicates that there are some sentences, not selected by the original method, which are considered to be important by both the Trigram Blocking and MMR-Select methods.

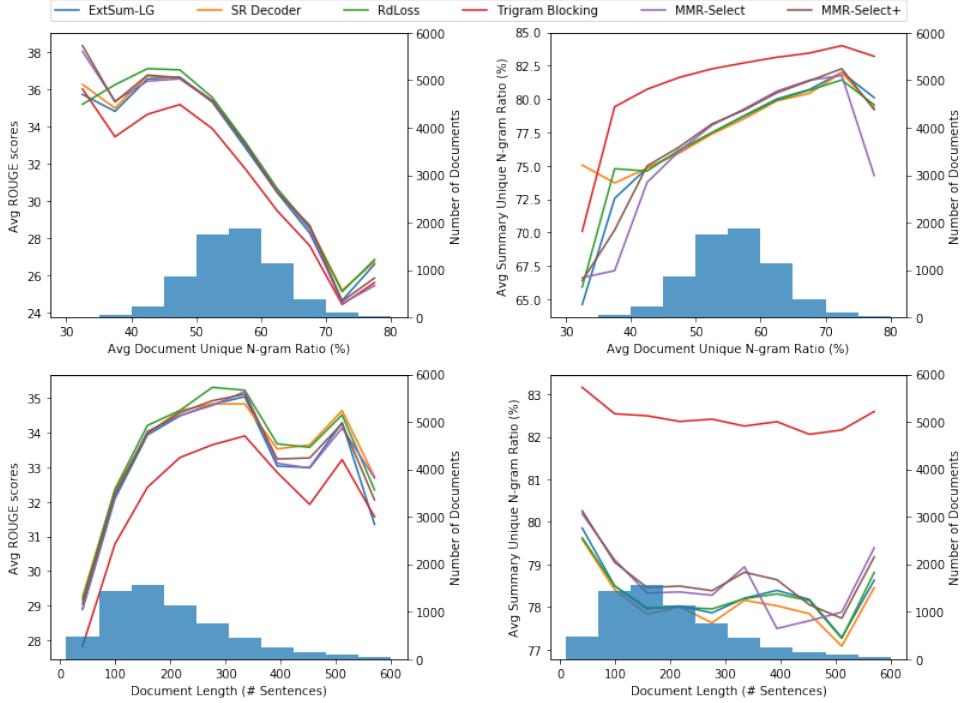


Figure 6: Comparing the average ROUGE scores and average unique n-gram ratios of different models on the arXiv dataset, conditioned on different degrees of redundancy and lengths of the document.<sup>8</sup>

-	ExtSumLG	+SR	+NeuSum	+RdLoss	+Tri-Block	+MMR-Select	+MMR-Select+
ExtSumLG	100.00	72.84	52.00	77.70	60.77	87.71	85.75
+SR	72.66	100.00	49.73	70.29	52.24	69.78	70.64
+Neusum	60.44	57.94	100.00	60.77	48.47	60.38	61.07
+RdLoss	80.84	73.32	54.40	100.00	57.67	79.03	80.08
+Tri-Block	64.85	55.89	44.51	59.15	100.00	64.72	64.38
+MMR-Select	90.49	72.17	53.59	78.37	62.56	100.00	91.15
+MMR-Select+	88.66	73.22	54.33	79.58	62.38	91.35	100.00
# Sent. Sel.	36979	36888	42981	38476	39463	38151	38236
# words/Sent	40.66	40.84	33.38	38.95	37.21	39.35	39.31

Table 5: Micro overlap ratio (%) between the selections of different methods and the total number and the average length of selected sentences in the test set of Pubmed.

### A.3 Analysis on Recall and Precision of ROUGE Scores

We also provide the Precision and Recall of the ROUGE scores in the main experiment, the results of Pubmed and arXiv datasets are shown in Table 7 and Table 8, respectively. It is interesting to see that the NeuSum Decoder tends to have a high precision but low recall, indicating that the generated summaries tend to be shorter and contain less useful information than the original method.

### A.4 Analysis on the Relative Position of Selections

We also show the relative position distribution of the selected sentences on both datasets in Figure 7 to verify if any redundancy reduction method has a

particular tendency to select sentences in particular position of the documents. However, as shown in the figure, the trends are all rather similar for all methods.

-	ExtSumLG	+SR	+NeuSum	+RdLoss	+Tri-Block	+MMR-Select	+MMR-Select+
ExtSumLG	100.00	72.06	-	76.51	56.22	75.04	80.21
+SR	73.84	100.00	-	69.07	49.16	62.82	67.03
+Neusum	-	-	-	-	-	-	-
+RdLoss	79.88	70.38	-	100.00	53.00	67.81	72.57
+Tri-Block	64.59	55.12	-	58.33	100.00	60.34	62.55
+MMR-Select	88.93	72.65	-	76.97	62.23	100.00	93.13
+MMR-Select+	89.96	73.36	-	77.96	61.06	88.14	100.00
# Sent. Sel.	39698	40681	-	41448	45611	47045	44526
# words/Sent	36.26	35.52	-	34.50	30.86	30.73	32.40

Table 6: Micro overlap ratio (%) between the selections of different methods and the total number and the average length of selected sentences in the test set of arXiv.

Categ.	Model	Pubmed					
		ROUGE-1		ROUGE-2		ROUGE-L	
		Prec.	Recall	Prec.	Recall	Prec.	Recall
C	Naive MMR	36.45	42.56	11.05	12.64	31.39	36.53
	ExtSum-LG <sup>9</sup>	44.05	51.08	19.82	22.71	39.74	45.97
A	+SR Decoder	44.00	51.10	19.75	22.68	39.66	45.96
A	+NeuSum Decoder	44.36	49.24	19.74	21.58	40.29	44.62
B	+RdLoss	44.30	51.09	20.11	22.88	40.09	46.11
C	+Trigram Blocking	42.67	48.54	17.51	19.73	38.45	43.64
C	+MMR-Select	44.25	51.09	19.98	22.75	40.08	46.07
C	+MMR-Select+	44.28	51.27	20.01	22.86	40.03	46.24

Table 7: Rouge Recall and Precision of different summarization models on the Pubmed dataset. **Green** numbers means it's **better** than ExtSum-LG on the certain metric, and the **red** numbers means worse.

Categ.	Model	Arxiv					
		ROUGE-1		ROUGE-2		ROUGE-L	
		Prec.	Recall	Prec.	Recall	Prec.	Recall
C	Naive MMR	29.61	42.69	7.45	10.78	24.92	35.82
	ExtSum-LG <sup>10</sup>	38.60	54.64	15.38	22.00	34.17	48.26
A	+SR Decoder	38.65	54.99	15.47	22.28	34.24	48.64
A	+NeuSum Decoder	-	-	-	-	-	-
B	+RdLoss	38.92	54.77	15.68	22.29	34.60	48.59
C	+Trigram Blocking	38.04	52.71	13.98	19.47	33.71	46.61
C	+MMR-Select	38.85	54.33	15.39	21.74	34.56	48.24
C	+MMR-Select+	38.75	54.67	15.41	21.96	34.44	48.51

Table 8: Rouge Recall and Precision of different summarization models on the Pubmed dataset. **Green** numbers means it's **better** than ExtSum-LG on the certain metric, and the **red** numbers means worse.

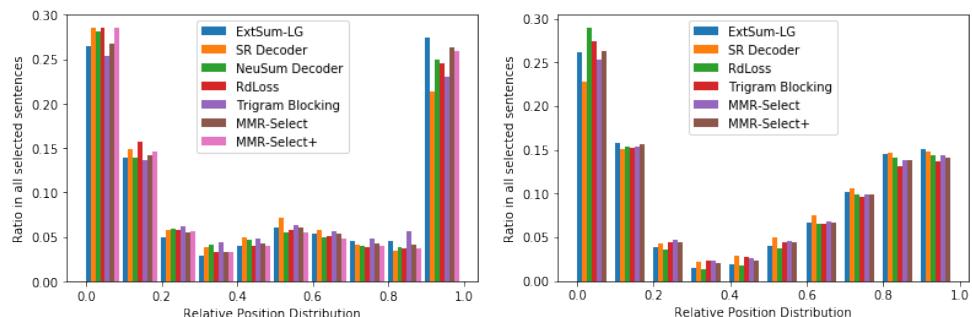


Figure 7: The relative position distribution of different redundancy reduction methods on Pubmed(left) and arXiv(right) datasets.

# A Cascade Approach to Neural Abstractive Summarization with Content Selection and Fusion

Logan Lebanoff,<sup>♦</sup> Franck Dernoncourt,<sup>◊</sup> Doo Soon Kim,<sup>◊</sup> Walter Chang,<sup>◊</sup> Fei Liu<sup>♦</sup>

<sup>♦</sup>Computer Science Department, University of Central Florida, Orlando, FL

<sup>◊</sup>Adobe Research, San Jose, CA

loganlebanoff@knights.ucf.edu {dernonco,dkim,wachang}@adobe.com  
feiliu@cs.ucf.edu

## Abstract

We present an empirical study in favor of a cascade architecture to neural text summarization. Summarization practices vary widely but few other than news summarization can provide a sufficient amount of training data enough to meet the requirement of end-to-end neural abstractive systems which perform content selection and surface realization jointly to generate abstracts. Such systems also pose a challenge to summarization evaluation, as they force content selection to be evaluated along with text generation, yet evaluation of the latter remains an unsolved problem. In this paper, we present empirical results showing that the performance of a cascaded pipeline that separately identifies important content pieces and stitches them together into a coherent text is comparable to or outranks that of end-to-end systems, whereas a pipeline architecture allows for flexible content selection. We finally discuss how we can take advantage of a cascaded pipeline in neural text summarization and shed light on important directions for future research.

## 1 Introduction

There is a variety of successful summarization applications but few can afford to have a large number of annotated examples that are sufficient to meet the requirement of end-to-end neural abstractive summarization. Examples range from summarizing radiology reports (Jing et al., 2019; Zhang et al., 2020) to congressional bills (Kornilova and Eidelman, 2019) and meeting conversations (Mehdad et al., 2013; Li et al., 2019; Koay et al., 2020). The lack of annotated resources suggests that end-to-end systems may not be a “one-size-fits-all” solution to neural text summarization. There is an increasing need to develop cascaded architectures to allow for customized content selectors to be combined with general-purpose neural text generators

to realize the full potential of neural abstractive summarization.

We advocate for explicit content selection as it allows for a rigorous evaluation and visualization of intermediate results of such a module, rather than associating it with text generation. Existing neural abstractive systems can perform content selection implicitly using end-to-end models (See et al., 2017; Celikyilmaz et al., 2018; Raffel et al., 2019; Lewis et al., 2020), or more explicitly, with an external module to select important sentences or words to aid generation (Tan et al., 2017; Gehrmann et al., 2018; Chen and Bansal, 2018; Kryściński et al., 2018; Hsu et al., 2018; Lebanoff et al., 2018, 2019b; Liu and Lapata, 2019). However, content selection concerns not only the selection of important segments from a document, but also the cohesiveness of selected segments and the amount of text to be selected in order for a neural text generator to produce a summary.

In this paper, we aim to investigate the feasibility of a cascade approach to neural text summarization. We explore a constrained summarization task, where an abstract is created one sentence at a time through a cascaded pipeline. Our pipeline architecture chooses one or two sentences from the source document, then highlights their summary-worthy segments and uses those as a basis for composing a summary sentence. When a pair of sentences are selected, it is important to ensure that they are *fusible*—there exists cohesive devices that tie the two sentences together into a coherent text—to avoid generating nonsensical outputs (Geva et al., 2019; Lebanoff et al., 2020). Highlighting sentence segments allows us to perform fine-grained content selection that guides the neural text generator to stitch selected segments into a coherent sentence. The contributions of this work are summarized as follows.

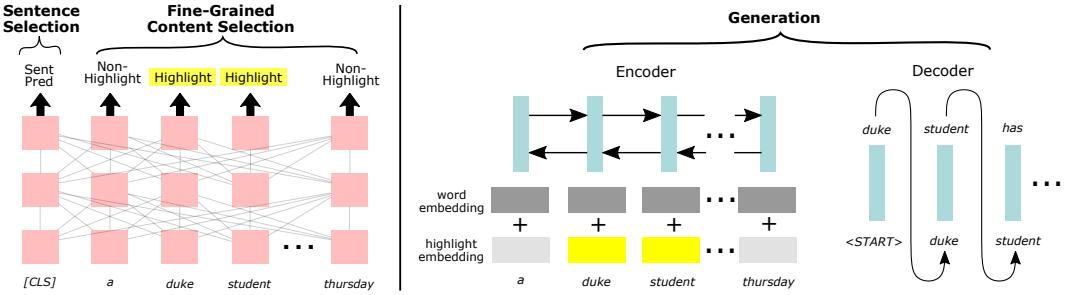


Figure 1: Model architecture. We divide the task between two main components: the first component performs sentence selection and fine-grained content selection, which are posed as a classification problem and a sequence-tagging problem, respectively. The second component receives the first component’s outputs as supplementary information to generate the summary. A cascade architecture provides the necessary flexibility to separate content selection from surface realization in abstractive summarization.

- We present an empirical study in favor of a cascade architecture for neural text summarization. Our cascaded pipeline chooses one or two sentences from the document and highlights their important segments; these segments are passed to a neural generator to produce a summary sentence.
- Our quantitative results show that the performance of a cascaded pipeline is comparable to or outranks that of end-to-end systems, with added benefit of flexible content selection. We discuss how we can take advantage of a cascade architecture and shed light on important directions for future research.<sup>1</sup>

## 2 A Cascade Approach

Our cascaded summarization approach focuses on shallow abstraction. It makes use of text transformations such as sentence shortening, paraphrasing and fusion (Jing and McKeown, 2000) and is in contrast to deep abstraction, where a full semantic analysis of the document is often required. A shallow approach helps produce abstracts that convey important information while, crucially, remaining faithful to the original. In what follows, we describe our approach to select single sentences and sentence pairs from the document, highlight summary-worthy segments and perform summary generation conditioned on highlights.

**Selection of Singletons and Pairs** Our approach iteratively selects one or two sentences from the input document; they serve as the basis for composing a single summary sentence. Previous research suggests that 60-85% of human-written summary

sentences are created by shortening a single sentence or merging a pair of sentences (Lebanoff et al., 2019b). We adopt this setting and present a coarse-to-fine strategy for content selection. Our strategy begins with selecting sentence singletons and pairs, followed by highlighting important segments of the sentences. Importantly, the strategy allows us to control which segments will be combined into a summary sentence—“compatible” segments come from either a single document sentence or a pair of *fusible* sentences. In contrast, when all important segments of the document are provided to a neural generator all at once (Gehrmann et al., 2018), it can happen that the generator arbitrarily stitches together text segments from unrelated sentences, yielding a summary that contains hallucinated content and fails to retain the meaning of the original document (Falke et al., 2019; Lebanoff et al., 2019a; Kryscinski et al., 2019).

We expect a sentence singleton or pair to be selected from the document if it contains salient content. Moreover, a pair of sentences should contain content that is compatible with each other. Given a sentence or pair of sentences from the document, our model predicts whether it is a valid instance to be compressed or merged to form a summary sentence. We follow (Lebanoff et al., 2019b) to use BERT (Devlin et al., 2019) to perform the classification. BERT is a natural choice since it takes one or two sentences and generates a classification prediction. It treats an input singleton or pair of sentences as a sequence of tokens. The tokens are fed to a series of Transformer block layers, consisting of multi-head self-attention modules. The first Transformer layer creates a contextual representation for each token, and each successive layer further refines those representations. An additional

<sup>1</sup>Our code is publicly available at <https://github.com/ucfnlp/cascaded-summ>

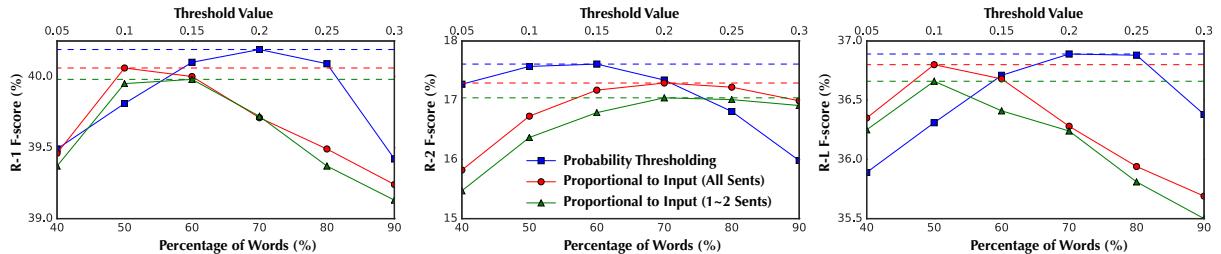


Figure 2: Comparison of various highlighting strategies. Thresholding obtains the best performance.

[CLS] token is added to contain the sentence representation. BERT is fine-tuned for our task by adding an output layer on top of the final layer representation  $\mathbf{h}_{[\text{CLS}]}^L$  for sequence  $s$ , as seen in Eq. (1).

$$p_{\text{sent}}(s) = \sigma(\mathbf{u}^\top \mathbf{h}_{[\text{CLS}]}^L) \quad (1)$$

where  $\mathbf{u}$  is a vector of weights and  $\sigma$  is the sigmoid function. The model predicts  $p_{\text{sent}}$  – whether the sentence singleton or pair is an appropriate one based on the [CLS] token representation. We describe the training data for this task in §3.

**Fine-Grained Content Selection** It is interesting to note that the previous architecture can be naturally extended to perform fine-grained content selection by highlighting important words of sentences. When two sentences are selected to generate a fusion sentence, it is desirable to identify segments of text from these sentences that are potentially compatible with each other. The coarse-to-fine method allows us to examine the intermediate results and compare them with ground-truth. Concretely, we add a classification layer to the final layer representation  $\mathbf{h}_i^L$  for each token  $w_i$  (Eq. (2)). The per-target-word loss is then interpolated with instance prediction (one or two sentences) loss using a coefficient  $\lambda$ . Such a multi-task learning objective has been shown to improve performance on a number of tasks (Guo et al., 2019).

$$p_{\text{highlight}}(w_i) = \sigma(\mathbf{v}^\top \mathbf{h}_i^L) \quad (2)$$

where  $\mathbf{v}$  is a vector of weights and  $\sigma$  is the sigmoid function. The model predicts  $p_{\text{highlight}}$  for each token – whether the token should be included in the output fusion, calculated based on the given token’s representation.

**Information Fusion** Given one or two sentences taken from a document and their fine-grained highlights, we proceed by describing a fusion process that generates a summary sentence from the selected content. Our model employs an encoder-decoder architecture based on pointer-generator

networks that has shown strong performance on its own and with adaptations (See et al., 2017; Gehrmann et al., 2018). We feed the sentence singleton or pair to the encoder along with highlights derived by the fine-grained content selector, the latter come in the form of binary tags. The tags are transformed to a “highlight-on” embedding for each token if it is chosen by the content selector, and a “highlight-off” embedding for each token not chosen. The highlight-on/off embeddings are added to token embeddings in an element-wise manner; both highlight and token embeddings are learned. An illustration is shown in Figure 1.

Highlights provide a valuable intermediate representation suitable for shallow abstraction. Our approach thus provides an alternative to methods that use more sophisticated representations such as syntactic/semantic graphs (Filippova and Strube, 2008; Banarescu et al., 2013; Liu et al., 2015). It is more straightforward to incorporate highlights into an encoder-decoder fusion model, and obtaining highlights through sequence tagging can be potentially adapted to new domains.

### 3 Experimental Results

**Data and Annotation** To enable direct comparison with end-to-end systems, we conduct experiments on the widely used CNN/DM dataset (See et al., 2017) to report results of our cascade approach. We use the procedure described in Lebanoff et al. (2019b) to create training instances for the sentence selector and fine-grained content selector. Our training data contains 1,053,993 instances; every instance contains one or two candidate sentences. It is a positive instance if a ground-truth summary sentence can be formed by compressing or merging sentences of the instance, negative otherwise. For positive instances, we highlight all lemmatized unigrams appearing in the summary, excluding punctuation. We further add smoothing to the labels by highlighting single words that con-

System	R-1	R-2	R-L
SumBasic (Vanderwende et al., 2007)	34.11	11.13	31.14
LexRank (Erkan and Radev, 2004)	35.34	13.31	31.93
Pointer-Generator (See et al., 2017)	39.53	17.28	36.38
FastAbsSum (Chen and Bansal, 2018)	40.88	17.80	<b>38.54</b>
BERT-Extr (Lebanoff et al., 2019b)	41.13	<b>18.68</b>	37.75
BottomUp (Gehrmann et al., 2018)	<b>41.22</b>	<b>18.68</b>	38.34
BERT-Abs (Lebanoff et al., 2019b)	37.15	15.22	34.60
Cascade-Fusion (Ours)	40.10	17.61	<b>36.71</b>
Cascade-Tag (Ours)	<b>40.24</b>	<b>18.33</b>	36.14
GT-Sent + Sys-Tag	50.40	27.74	46.25
GT-Sent + Sys-Tag + Fusion	51.33	28.08	47.50
GT-Sent + GT-Tag	<b>74.80</b>	48.21	<b>67.40</b>
GT-Sent + GT-Tag + Fusion	72.70	<b>48.33</b>	67.06

(SYSTEM SENTS) A Duke student has admitted to hanging a noose made of rope from a tree near a student union, university officials said Thursday. The student was identified during an investigation by campus police and the office of student affairs and admitted to placing the noose on the tree early Wednesday, the university said.
(CASCADE-FUSION) A Duke student was identified during an investigation by campus police and the office of student affairs and admitted to placing the noose on the tree early Wednesday.
(GT SENTS) In a news release, it said the student was no longer on campus and will face student conduct review. Duke University is a private college with about 15,000 students in Durham, North Carolina.
(GT SENTS + FUSION) Duke University student was no longer on campus and will face student conduct review.
(REFERENCE) Student is no longer on Duke University campus and will face disciplinary review.

Table 1: (LEFT) Summarization results on CNN/DM test set. Our cascade approach performs comparable to strong extractive and abstractive baselines; oracle models using ground-truth sentences and segment highlights perform the best. (RIGHT) Example source sentences and their fusions. Dark highlighting is content taken from the first sentence, and light highlighting comes from the second. Our *Cascade-Fusion* approach effectively performs entity replacement by replacing “student” in the second sentence with “a Duke student” from the first sentence.

nect two highlighted phrases and by dehighlighting isolated stopwords. At test time, four highest-scored instances are selected per document; their important segments are highlighted by content selector then passed to the fusion step to produce a summary sentence each. The hyperparameter  $\lambda$  for weighing the per-target-word loss is set to 0.2 and highlighting threshold value is 0.15. The model hyperparameters are tuned on the validation split.

**Summarization Results** We show experimental results on the standard test set and evaluated by ROUGE metrics (Lin, 2004) in Table 1. The performance of our cascade approaches, *Cascade-Fusion* and *Cascade-Tag*, is comparable to or outranks a number of extractive and abstractive baselines. Particularly, *Cascade-Tag* does not use a fusion step (§2) and is the output of fine-grained content selection. *Cascade-Fusion* provides a direct comparison against BERT-Abs (Lebanoff et al., 2019b) that uses sentence selection and fusion but lacks a fine-grained content selector.

Our results suggest that a coarse-to-fine content selection strategy remains necessary to guide the fusion model to produce informative sentences. We observe that the addition of the fusion model has only a moderate impact on ROUGE scores, but the fusion process can reorder text segments to create true and grammatical sentences, as shown in Table 1. We analyze the performance of a number of oracle models that use ground-truth sentence selection (GT-Sent) and tagging (GT-Tag). When given ground-truth sentences as input, our cascade

models achieve  $\sim$ 10 points of improvement in all ROUGE metrics. When the models are also given ground-truth highlights, they achieve an additional 20 points of improvement. In a preliminary examination, we observe that not all highlights are included in the summary during fusion, indicating there is space for improvement. These results show that cascade architectures have great potential to generate shallow abstracts and future emphasis may be placed on accurate content selection.

**How much should we highlight?** It is important to quantify the amount of highlighting required for generating a summary sentence. Highlighting too much or too little can be unhelpful. We experiment with three methods to determine the appropriate amount of words to highlight. *Probability Thresholding* chooses a set threshold whereby all words that have a probability higher than the threshold are highlighted. When *Proportional to Input* is used, the highest probability words are iteratively highlighted until a target rate is reached. The amount of highlighting can be proportional to the total number of words per instance (one or two sentences) or per document, containing all sentences selected for the document.

We investigate the effect of varying the amount of highlighting in Figure 2. Among the three methods, probability thresholding performs the best, as it gives more freedom to content selection. If the model scores all of the words in sentences highly, then we should correspondingly highlight all of the words. If only very few words score highly, then

we should only pick those few.

Highlighting a certain percentage of words tend to perform less well. On our dataset, a threshold value of 0.15–0.20 produces the best ROUGE scores. Interestingly, these thresholds end up highlighting 58–78% of the words of each sentence. Compared to what the generator was trained on, which had a median of 31% of each sentence highlighted, the system’s rate of highlighting is higher. If the model’s highlighting rate is set to be similar to that of the ground-truth, it yields much lower ROUGE scores (cf. threshold value of 0.3 in Figure 2). This observation suggests that the amount of highlighting can be related to the effectiveness of content selector and it may be better to highlight more than less.

## 4 Conclusion

We present a cascade approach to neural abstractive summarization that separates content selection from surface realization. Importantly, our approach makes use of text highlights as intermediate representation; they are derived from one or two sentences using a coarse-to-fine content selection strategy, then passed to a neural text generator to compose a summary sentence. A successful cascade approach is expected to accurately select sentences and highlight an appropriate amount of text, both can be customized for domain-specific tasks.

## Acknowledgments

We are grateful to the anonymous reviewers for their comments and suggestions. This research was supported in part by the National Science Foundation grant IIS-1909603.

## References

- Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. *Abstract Meaning Representation for sembanking*. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.
- Asli Celikyilmaz, Antoine Bosselut, Xiaodong He, and Yejin Choi. 2018. *Deep communicating agents for abstractive summarization*. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1662–1675, New Orleans, Louisiana. Association for Computational Linguistics.
- Yen-Chun Chen and Mohit Bansal. 2018. *Fast abstractive summarization with reinforce-selected sentence rewriting*. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 675–686, Melbourne, Australia. Association for Computational Linguistics.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- Günes Erkan and Dragomir R. Radev. 2004. *LexRank: Graph-based Lexical Centrality as Salience in Text Summarization*. *Journal of Artificial Intelligence Research*.
- Tobias Falke, Leonardo F. R. Ribeiro, Prasetya Ajie Utama, Ido Dagan, and Iryna Gurevych. 2019. *Ranking generated summaries by correctness: An interesting but challenging application for natural language inference*. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2214–2220, Florence, Italy. Association for Computational Linguistics.
- Katja Filippova and Michael Strube. 2008. *Sentence fusion via dependency graph compression*. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 177–185, Honolulu, Hawaii. Association for Computational Linguistics.
- Sebastian Gehrmann, Yuntian Deng, and Alexander Rush. 2018. *Bottom-up abstractive summarization*. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4098–4109, Brussels, Belgium. Association for Computational Linguistics.
- Mor Geva, Eric Malmi, Idan Szpektor, and Jonathan Berant. 2019. *DiscoFuse: A large-scale dataset for discourse-based sentence fusion*. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3443–3455, Minneapolis, Minnesota. Association for Computational Linguistics.
- Han Guo, Ramakanth Pasunuru, and Mohit Bansal. 2019. *AutoSeM: Automatic task selection and mixing in multi-task learning*. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3520–3531, Minneapolis, Minnesota. Association for Computational Linguistics.

- Wan-Ting Hsu, Chieh-Kai Lin, Ming-Ying Lee, Kerui Min, Jing Tang, and Min Sun. 2018. **A unified model for extractive and abstractive summarization using inconsistency loss.** In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 132–141, Melbourne, Australia. Association for Computational Linguistics.
- Baoyu Jing, Zeya Wang, and Eric Xing. 2019. **Show, describe and conclude: On exploiting the structure information of chest x-ray reports.** In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6570–6580, Florence, Italy. Association for Computational Linguistics.
- Hongyan Jing and Kathleen R. McKeown. 2000. **Cut and paste based text summarization.** In *1st Meeting of the North American Chapter of the Association for Computational Linguistics*.
- Jia Jin Koay, Alexander Roustai, Xiaojin Dai, Alec Dillon, and Fei Liu. 2020. How domain terminology affects meeting summarization performance. In *Proceedings of the 28th International Conference on Computational Linguistics (COLING)*.
- Anastassia Kornilova and Vladimir Eidelman. 2019. **BillSum: A corpus for automatic summarization of US legislation.** In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 48–56, Hong Kong, China. Association for Computational Linguistics.
- Wojciech Kryscinski, Nitish Shirish Keskar, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. **Neural text summarization: A critical evaluation.** In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 540–551, Hong Kong, China. Association for Computational Linguistics.
- Wojciech Kryscinski, Romain Paulus, Caiming Xiong, and Richard Socher. 2018. **Improving abstraction in text summarization.** In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1808–1817, Brussels, Belgium. Association for Computational Linguistics.
- Logan Lebanoff, John Muchovej, Franck Dernoncourt, Doo Soon Kim, Seokhwan Kim, Walter Chang, and Fei Liu. 2019a. **Analyzing sentence fusion in abstractive summarization.** In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 104–110, Hong Kong, China. Association for Computational Linguistics.
- Logan Lebanoff, John Muchovej, Franck Dernoncourt, Doo Soon Kim, Lidan Wang, Walter Chang, and Fei Liu. 2020. **Understanding points of correspondence between sentences for abstractive summarization.** In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, Seattle, United States. Association for Computational Linguistics.
- Logan Lebanoff, Kaiqiang Song, Franck Dernoncourt, Doo Soon Kim, Seokhwan Kim, Walter Chang, and Fei Liu. 2019b. **Scoring sentence singletons and pairs for abstractive summarization.** In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2175–2189, Florence, Italy. Association for Computational Linguistics.
- Logan Lebanoff, Kaiqiang Song, and Fei Liu. 2018. **Adapting the neural encoder-decoder framework from single to multi-document summarization.** In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4131–4141, Brussels, Belgium. Association for Computational Linguistics.
- Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. **BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.** In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.
- Manling Li, Lingyu Zhang, Heng Ji, and Richard J. Radke. 2019. **Keep meeting summaries on topic: Abstractive multi-modal meeting summarization.** In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2190–2196, Florence, Italy. Association for Computational Linguistics.
- Chin-Yew Lin. 2004. **ROUGE: A package for automatic evaluation of summaries.** In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.
- Fei Liu, Jeffrey Flanigan, Sam Thomson, Norman Sadeh, and Noah A. Smith. 2015. **Toward abstractive summarization using semantic representations.** In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1077–1086, Denver, Colorado. Association for Computational Linguistics.
- Yang Liu and Mirella Lapata. 2019. **Hierarchical transformers for multi-document summarization.** In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5070–5081, Florence, Italy. Association for Computational Linguistics.
- Yashar Mehdad, Giuseppe Carenini, Frank Tompa, and Raymond T. Ng. 2013. **Abstractive meeting summarization with entailment and fusion.** In *Proceedings of the 14th European Workshop on Natural Language Generation*, pages 136–146, Sofia, Bulgaria. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv:1910.10683*.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get To The Point: Summarization with Pointer-Generator Networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. 2017. Abstractive document summarization with a graph-based attentional neural model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1171–1181, Vancouver, Canada. Association for Computational Linguistics.

Lucy Vanderwende, Hisami Suzuki, Chris Brockett, and Ani Nenkova. 2007. Beyond SumBasic: Task-focused Summarization with Sentence Simplification and Lexical Expansion. *Information Processing and Management*, 43(6):1606–1618.

Yuhao Zhang, Derek Merck, Emily Bao Tsai, Christopher D. Manning, and Curtis P. Langlotz. 2020. Optimizing the factual correctness of a summary: A study of summarizing radiology reports. In *Proceedings of the 58th Annual Conference of the Association for Computational Linguistics (ACL)*.

# Mixed-Lingual Pre-training for Cross-lingual Summarization

Ruochen Xu\*, Chenguang Zhu\*, Yu Shi, Michael Zeng, Xuedong Huang

Microsoft Cognitive Services Research Group

{ruox, chezhu, yushi, nzeng, xdh}@microsoft.com

## Abstract

Cross-lingual Summarization (CLS) aims at producing a summary in the target language for an article in the source language. Traditional solutions employ a two-step approach, i.e. translate→summarize or summarize→translate. Recently, end-to-end models have achieved better results, but these approaches are mostly limited by their dependence on large-scale labeled data. We propose a solution based on mixed-lingual pre-training that leverages both cross-lingual tasks such as translation and monolingual tasks like masked language models. Thus, our model can leverage the massive monolingual data to enhance its modeling of language. Moreover, the architecture has no task-specific components, which saves memory and increases optimization efficiency. We show in experiments that this pre-training scheme can effectively boost the performance of cross-lingual summarization. In Neural Cross-Lingual Summarization (NCLS) (Zhu et al., 2019b) dataset, our model achieves an improvement of 2.82 (English to Chinese) and 1.15 (Chinese to English) ROUGE-1 scores over state-of-the-art results.

## 1 Introduction

Text summarization can facilitate the propagation of information by providing an abridged version for long articles and documents. Meanwhile, the globalization progress has prompted a high demand of information dissemination across language barriers. Thus, the cross-lingual summarization (CLS) task emerges to provide accurate gist of articles in a foreign language.

Traditionally, most CLS methods follow the two-step pipeline approach: either translate the article into the target language and then summarize it (Leuski et al., 2003), or summarize the article in the source language and then translate it (Wan

et al., 2010). Although this method can leverage off-the-shelf summarization and MT models, it suffers from error accumulation from two independent subtasks. Therefore, several end-to-end approaches have been proposed recently (Zhu et al., 2019b; Ouyang et al., 2019; Duan et al., 2019), which conduct both translation and summarization simultaneously. Easy to optimize as these methods are, they typically require a large amount of cross-lingual summarization data, which may not be available especially for low-resource languages. For instance, NCLS (Zhu et al., 2019b) proposes to co-train on monolingual summarization (MS) and machine translation (MT) tasks, both of which require tremendous labeling efforts.

On the other hand, the pre-training strategy has proved to be very effective for language understanding (Devlin et al., 2018; Holtzman et al., 2019) and cross-lingual learning (Lample and Conneau, 2019; Chi et al., 2019). One of the advantages of pre-training is that many associated tasks are self-learning by nature, which means no labeled data is required. This greatly increases the amount of training data exposed to the model, thereby enhancing its performance on downstream tasks.

Therefore, we leverage large-scale pre-training to improve the quality of cross-lingual summarization. Built upon a transformer-based encoder-decoder architecture (Vaswani et al., 2017), our model is pre-trained on both monolingual tasks including masked language model (MLM), denoising autoencoder (DAE) and monolingual summarization (MS), and cross-lingual tasks such as cross-lingual masked language model (CMLM) and machine translation (MT). This mixed-lingual pre-training scheme can take advantage of massive unlabeled monolingual data to improve the model’s language modeling capability, and leverage cross-lingual tasks to improve the model’s cross-lingual representation. We then finetune the model on the

\* Equal contribution

downstream cross-lingual summarization task.

Furthermore, based on a shared multi-lingual vocabulary, our model has a shared encoder-decoder architecture for all pre-training and finetuning tasks, whereas NCLS (Zhu et al., 2019b) sets aside task-specific decoders for machine translation, monolingual summarization, and cross-lingual summarization.

In the experiments, our model outperforms various baseline systems on the benchmark dataset NCLS (Zhu et al., 2019b). For example, our model achieves 3.27 higher ROUGE-1 score in Chinese to English summarization than the state-of-the-art result and 1.28 higher ROUGE-1 score in English to Chinese summarization. We further conduct an ablation study to show that each pretraining task contributes to the performance, especially our proposed unsupervised pretraining tasks.

## 2 Related Work

### 2.1 Pre-training

Pre-training language models (Devlin et al., 2018; Dong et al., 2019) have been widely used in NLP applications such as question answering (Zhu et al., 2018), sentiment analysis (Peters et al., 2018), and summarization (Zhu et al., 2019a; Yang et al., 2020). In multi-lingual scenarios, recent works take input from multiple languages and shows great improvements on cross-lingual classification (Lample and Conneau, 2019; Pires et al., 2019; Huang et al., 2019) and unsupervised machine translation (Liu et al., 2020). Artetxe and Schwenk (2019) employs the sequence encoder from a machine translation model to produce cross-lingual sentence embeddings. Chi et al. (2019) uses multi-lingual pre-training to improve cross-lingual question generation and zero-shot cross-lingual summarization. Their model trained on articles and summaries in one language is directly used to produce summaries for articles in another language, which is different from our task of producing summaries of one language for an article from a foreign language.

### 2.2 Cross-lingual Summarization

Early literatures on cross-lingual summarization focus on the two-step approach involving machine translation and summarization (Leuski et al., 2003; Wan et al., 2010), which often suffer from error propagation issues due to the imperfect modular systems. Recent end-to-end deep learning models have greatly enhanced the performance. Shen et al.

(2018) presents a solution to zero-shot cross-lingual headline generation by using machine translation and summarization datasets. Duan et al. (2019) leverages monolingual abstractive summarization to achieve zero-shot cross-lingual abstractive sentence summarization. NCLS (Zhu et al., 2019b) proposes a cross-lingual summarization system for large-scale datasets for the first time. It uses multi-task supervised learning and shares the encoder for monolingual summarization, cross-lingual summarization, and machine translation. However, each of these tasks requires a separate decoder. In comparison, our model shares the entire encoder-decoder architecture among all pre-training and finetuning tasks, and leverages unlabeled data for monolingual masked language model training. A concurrent work by Zhu et al. (2020) improves the performance by combining the neural model with an external probabilistic bilingual lexicon.

## 3 Method

### 3.1 Pre-training Objectives

We propose a set of multi-task pre-training objectives on both monolingual and cross-lingual corpus. For monolingual corpus, we use the masked language model (MLM) from Raffel et al. (2019). The input is the original sentence masked by sentinel tokens, and the target is the sequence consists of each sentinel token followed by the corresponding masked token. The other monolingual task is the denoising auto-encoder (DAE), where the corrupted input is constructed by randomly dropping, masking, and shuffling a sentence and the target is the original sentence. Since our final task is summarization, we also include monolingual summarization (MS) as a pre-training task.

To leverage cross-lingual parallel corpus, we introduce the cross-lingual masked language model (CMLM). CMLM is an extension of MLM on the parallel corpus. The input is the concatenation of a sentence in language A and its translation in language B. We then randomly select one sentence and mask some of its tokens by sentinels. The target is to predict the masked tokens in the same way as MLM. Different from MLM, the masked tokens in CMLM are predicted not only from the context within the same language but also from their translations in another language, which encourages the model to learn language-invariant representations. Note that CMLM is similar to the Translation Language Model (TLM) loss proposed in Lample

Objective	Supervised	Multi-lingual	Inputs	Targets
Masked Language Model			France <X> Morocco in <Y> exhibition match.	<X> beats <Y> an
Denoising Auto-Encoder			France beats <M> in <M> exhibition .	France beats Morocco in an exhibition match.
Monolingual Summarization	✓		World champion France overcame a stuttering start to beat Morocco 1-0 in a scrappy exhibition match on Wednesday night.	France beats Morocco in an exhibition match.
Cross-lingual MLM	✓	✓	France <X> Morocco in <Y> exhibition match. 法国队在一场比赛中击败摩洛哥队。	<X> beats <Y> an
Cross-lingual MLM	✓	✓	France beats Morocco in an exhibition match. <X>队在一场比赛中<Y>摩洛哥队。	<X>法国<Y>击败
Machine Translation	✓	✓	France beats Morocco in an exhibition match.	法国队在一场比赛中击败摩洛哥队。

Table 1: Examples of inputs and targets used by different objectives for the sentence “France beats Morocco in an exhibition match” with its Chinese translation. We use  $\langle X \rangle$  and  $\langle Y \rangle$  to denote sentinel tokens and  $\langle M \rangle$  to denote shared mask tokens.

and Conneau (2019). The key differences are: 1) TLM randomly masks tokens in sentences from both languages, while CMLM only masks tokens from one language; 2) TLM is applied on encoder-only networks while we employ CMLM on the encoder-decoder network. In addition to CMLM, we also include standard machine translation (MT) objective, in which the input and output are the unchanged source and target sentences, respectively.

The examples of inputs and targets used by our pre-training objectives are shown in Table 1.

### 3.2 Unified Model for Pre-training and Finetuning

While NCLS (Zhu et al., 2019b) uses different decoders for various pre-training objectives, we employ a unified Transformer (Vaswani et al., 2017) encoder-decoder model for all pre-training and finetuning tasks. This makes our model learn a cross-lingual representation efficiently. A shared dictionary across all languages is used. To accommodate multi-task and multilingual objectives, we introduce language id symbols to indicate the target language, and task symbols to indicate the target task. For instance, for the CMLM objective where the target language is Chinese, the decoder takes  $\langle \text{cmlm} \rangle$  and  $\langle \text{zh} \rangle$  as the first two input tokens. We empirically find that our model does not suffer from the phenomenon of forgetting target language controllability as in Chi et al. (2019), which requires manual freezing of encoder or decoder during finetuning. After pretraining, we conduct finetuning on cross-lingual summarization data.

## 4 Experiments

### 4.1 Dataset

We conduct our experiment on NCLS dataset (Zhu et al., 2019b), which contains paired data of English articles with Chinese summaries, and Chinese articles with English summaries. The cross-lingual training data is automatically generated by a machine translation model. For finetuning and testing, we followed the same train/valid/test split of the original dataset. We refer readers to Table 1 in Zhu et al. (2019b) for detailed statistics of the dataset.

For pre-training, we obtain monolingual data for English and Chinese from the corresponding Wikipedia dump. There are 83 million sentences for English monolingual corpus and 20 million sentences for Chinese corpus. For parallel data between English and Chinese, we use the parallel corpus from Lample and Conneau (2019), which contains 9.6 million paired sentences. For monolingual summarization objective, we use CNN/DailyMail dataset (Nallapati et al., 2016) for English summarization and LCSTS dataset (Hu et al., 2015) for Chinese summarization.

### 4.2 Implementation Details

Our transformer model has 6 layers and 8 heads in attention. The input and output dimensions  $d_{model}$  for all transformer blocks are 512 and the inner dimension  $d_{ff}$  is 2048.

We use a dropout probability of 0.1 on all layers. We build a shared SentencePiece (Kudo and Richardson, 2018) vocabulary of size 33,000 from a balanced mix of the monolingual Wikipedia corpus. The model has approximately 61M parameters.

For MLM we use a mask probability of 0.15. For DAE, we set both the mask and drop out rate

	English→Chinese			Chinese→English		
	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-1	ROUGE-2	ROUGE-L
TETran	26.15	10.60	23.24	23.09	7.33	18.74
GETran	28.19	11.40	25.77	24.34	9.14	20.13
TLTran	30.22	12.20	27.04	33.92	15.81	29.86
GLTran	32.17	13.85	29.43	35.45	16.86	31.28
NCLS	36.82	18.72	33.20	38.85	21.93	35.05
NCLS-MS	38.25	20.20	34.76	40.34	22.65	36.39
NCLS-MT	40.23	22.32	36.59	40.25	22.58	36.21
XNLG	39.85	24.47	28.28	38.34	19.65	33.66
ATS	40.68	24.12	<b>36.97</b>	40.47	22.21	36.89
Ours	<b>43.50</b>	<b>25.41</b>	29.66	<b>41.62</b>	<b>23.35</b>	<b>37.26</b>

Table 2: ROUGE-1, ROUGE-2, ROUGE-L for English to Chinese and Chinese to English summarization on NCLS dataset.

to 0.1. For all pre-training and finetuning we use RAdam optimizer (Liu et al., 2019) with  $\beta_1 = 0.9$ ,  $\beta_2 = 0.999$ . The initial learning rate is set to  $10^{-9}$  for pre-training and  $10^{-4}$  for finetuning. The learning rate is linearly increased to 0.001 with 16,000 warmup steps followed by an exponential decay. For decoding, we use a beam size of 6 and a maximum generation length of 200 tokens for all experiments.

	English→Chinese		
	ROUGE-1	ROUGE-2	ROUGE-L
Ours	43.50	25.41	29.66
- MS	42.48	24.45	28.49
- MT	42.12	23.97	28.74
- MLM, DAE	41.82	23.85	28.40
- All Pretraining	41.12	23.67	28.53

Table 3: Finetuning performance on English→Chinese summarization starting with various ablated pre-trained models.

### 4.3 Baselines

We first include a set of pipeline methods from Zhu et al. (2019b) which combines monolingual summarization and machine translation. **TETran** first translates the source document and then uses LexRank (Erkan and Radev, 2004) to summarize the translated document. **TLTran** first summarizes the source document and then translates the summary. **GETran** and **GLTran** replace the translation model in TETran and TLTran with Google Translator<sup>1</sup> respectively.

We also include three strong baselines from Zhu et al. (2019b): **NCLS**, **NCLS-MS** and **NCLS-MT**.

<sup>1</sup><https://translate.google.com/>

NCLS trains a standard Transformer model on the cross-lingual summarization dataset. NCLS-MS and NCLS-MT both use one encoder and multiple decoders for multi-task scenarios. NCLS-MS combines the cross-lingual summarization task with monolingual summarization while NCLS-MT combines it with machine translation.

We finetune **XNLG** model from Chi et al. (2019) on the same cross-lingual summarization data. We finetune all layers of XNLG in the same way as our pretrained model.

Finally, we include the result of **ATS** from the concurrent work of Zhu et al. (2020).

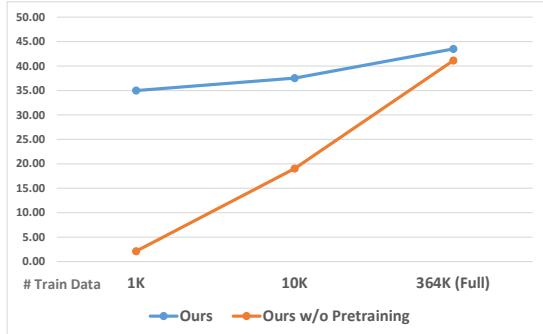
### 4.4 Results

Table 2 shows the ROUGE scores of generated summaries in English-to-Chinese and Chinese-to-English summarization. As shown, pipeline models, although incorporating state-of-the-art machine translation systems, achieve sub-optimal performance in both directions, proving the advantages of end-to-end models.

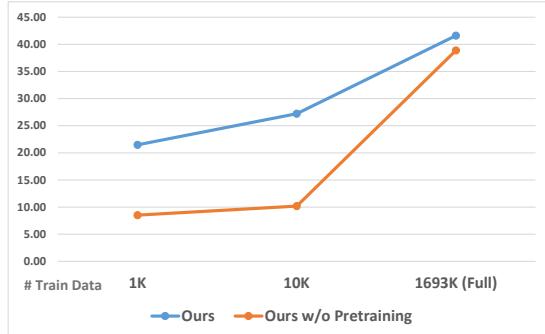
Our model outperforms all baseline models in all metrics except for ROUGE-L in English-to-Chinese. For instance, our model achieves 2.82 higher ROUGE-1 score in Chinese to English summarization than the previously best result and 1.15 higher ROUGE-1 score in English to Chinese summarization, which shows the effectiveness of utilizing multilingual and multi-task data to improve cross-lingual summarization.

### 4.5 Ablation Study

Table 3 shows the ablation study of our model on English to Chinese summarization. We remove



(a) English→Chinese ROUGE-1



(b) Chinese→English ROUGE-1

Figure 1: ROUGE-1 performance on NCLS dataset when the cross-lingual summarization training data is subsampled to size of 1k and 10k. The result on the full dataset is also shown.

from the pre-training objectives i) all monolingual unsupervised tasks (MLM, DAE), ii) machine translation (MT), iii) monolingual summarization (MS), and iv) all the objectives. Note that ”- All Pretraining” and NCLS both only train on the cross-lingual summarization data. The performance difference between the two is most likely due to the difference in model size, vocabulary, and other hyperparameters.

As shown, the pre-training can improve ROUGE-1, ROUGE-2, and ROUGE-L by 2.38, 1.74, and 1.13 points respectively on Chinese-to-English summarization. Moreover, all pre-training objectives have various degrees of contribution to the results, and the monolingual unsupervised objectives (MLM and DAE) are relatively the most important. This verifies the effectiveness of leveraging unsupervised data in the pre-training.

**Low-resource scenario.** We sample subsets of size 1K and 10K from the training data of cross-lingual summarization and finetune our pre-trained model on those subsets. Figure 1 shows the the performance of the pre-trained model and the model trained from scratch on the same subsets. As shown, the gain from pre-training is larger when the size of training data is relatively small. This proves the effectiveness of our approach to deal with low-resource language in cross-lingual summarization.

## 5 Conclusion

We present a mix-lingual pre-training model for cross-lingual summarization. We optimize a shared encoder-decoder architecture for multi-lingual and multi-task objectives. Experiments on a benchmark

dataset show that our model outperforms pipeline-based and other end-to-end baselines. Through an ablation study, we show that all pretraining objectives contribute to the model’s performance.

## References

- Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.
- Zewen Chi, Li Dong, Furu Wei, Wenhui Wang, Xian-Ling Mao, and Heyan Huang. 2019. Cross-lingual natural language generation via pre-training. *arXiv preprint arXiv:1909.10481*.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
- Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems*, pages 13063–13075.
- Xiangyu Duan, Mingming Yin, Min Zhang, Boxing Chen, and Weihua Luo. 2019. Zero-shot cross-lingual abstractive sentence summarization through teaching generation and attention. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3162–3172.
- Günes Erkan and Dragomir Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*, 22:457–479.

- Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.
- Baotian Hu, Qingcai Chen, and Fangze Zhu. 2015. Lsts: A large scale chinese short text summarization dataset. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1967–1972.
- Haoyang Huang, Yaobo Liang, Nan Duan, Ming Gong, Linjun Shou, Daxin Jiang, and Ming Zhou. 2019. Unicoder: A universal language encoder by pre-training with multiple cross-lingual tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2485–2494.
- Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.
- Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*.
- Anton Leuski, Chin-Yew Lin, Liang Zhou, Ulrich Germann, Franz Josef Och, and Eduard Hovy. 2003. Cross-lingual c\* st\* rd: English access to hindi information. *ACM Transactions on Asian Language Information Processing (TALIP)*, 2(3):245–269.
- Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. 2019. On the variance of the adaptive learning rate and beyond. In *International Conference on Learning Representations*.
- Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *arXiv preprint arXiv:2001.08210*.
- Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Caglar Gulcehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290.
- Jessica Ouyang, Boya Song, and Kathleen McKeown. 2019. A robust abstractive system for cross-lingual summarization. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2025–2031.
- Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.
- Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual bert? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.
- Shi-qi Shen, Yun Chen, Cheng Yang, Zhi-yuan Liu, Mao-song Sun, et al. 2018. Zero-shot cross-lingual neural headline generation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(12):2319–2327.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.
- Xiaojun Wan, Huiying Li, and Jianguo Xiao. 2010. Cross-language document summarization based on machine translation quality prediction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 917–926. Association for Computational Linguistics.
- Ziyi Yang, Chenguang Zhu, Robert Gmyr, Michael Zeng, Xuedong Huang, and Eric Darve. 2020. Ted: A pretrained unsupervised summarization model with theme modeling and denoising. *arXiv preprint arXiv:2001.00725*.
- Chenguang Zhu, Ziyi Yang, Robert Gmyr, Michael Zeng, and Xuedong Huang. 2019a. Make lead bias in your favor: A simple and effective method for news summarization. *arXiv preprint arXiv:1912.11602*.
- Chenguang Zhu, Michael Zeng, and Xuedong Huang. 2018. Sdnet: Contextualized attention-based deep network for conversational question answering. *arXiv preprint arXiv:1812.03593*.
- Junnan Zhu, Qian Wang, Yining Wang, Yu Zhou, Jiajun Zhang, Shaonan Wang, and Chengqing Zong. 2019b. Ncls: Neural cross-lingual summarization. *arXiv preprint arXiv:1909.00156*.
- Junnan Zhu, Yu Zhou, Jiajun Zhang, and Chengqing Zong. 2020. Attend, translate and summarize: An efficient method for neural cross-lingual summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1309–1321.

# Point-of-Interest Oriented Question Answering with Joint Inference of Semantic Matching and Distance Correlation

**Yifei Yuan**

CUHK

HongKong SAR

yfyuan@se.cuhk.edu.hk

**Jingbo Zhou**

Baidu Inc.

China

zhoujingbo@baidu.com

**Wai Lam**

CUHK

HongKong SAR

wlam@se.cuhk.edu.hk

## Abstract

Point-of-Interest (POI) oriented question answering (QA) aims to return a list of POIs given a question issued by a user. Recent advances in intelligent virtual assistants have opened the possibility of engaging the client software more actively in the provision of location-based services, thereby showing great promise for automatic POI retrieval. Some existing QA methods can be adopted on this task such as QA similarity calculation and semantic parsing using pre-defined rules. The returned results, however, are subject to inherent limitations due to the lack of the ability for handling some important POI related information, including tags, location entities, and proximity-related terms (e.g. “nearby”, “close”). In this paper, we present a novel deep learning framework integrated with joint inference to capture both tag semantic and geographic correlation between question and POIs. One characteristic of our model is to propose a special cross attention question embedding neural network structure to obtain question-to-POI and POI-to-question information. Besides, we utilize a skewed distribution to simulate the spatial relationship between questions and POIs. By measuring the results offered by the model against existing methods, we demonstrate its robustness and practicability, and supplement our conclusions with empirical evidence.

## 1 Introduction

Point-of-Interest (POI) oriented question answering (QA) problem is a special QA task which aims to answer users’ questions by generating a list of POIs. With the rapid development of smart agents (e.g. Amazon Echo) and intelligent virtual assistants (e.g. Apple Siri and Google Assistant), there are many POI oriented queries being requested everyday. Some examples of these questions are “Where can I take my kid to have fun nearby New York City” or “Where can we go in LA with my friends”. The answers are typically a list of POIs

such as parks, malls, or restaurants corresponding to the details provided by users in the questions. According to some statistics, there are millions of POI oriented QA questions being requested per day on a mobile search engine in China.

Generally speaking, semantic parsing and similarity matching methods are utilized to tackle the POI oriented QA problem in current solutions. Nevertheless, both of them are subject to inherent limitations and deserve to be improved. Semantic parsing based methods convert the questions to formal representations (such as SQL queries) using pre-defined rules, then get the POI results from the query. Difficulties arise, however, when the form of the questions varies from person to person. Besides, due to ambiguous expressions of a specific tag, semantic parsing methods always fail to match mentioned tags to the POI database. Furthermore, based only on tag information, it is almost impossible for semantic parsing methods to make use of the distance correlation between questions and POIs.

Another line of solution is adopting similarity matching models for calculating the similarity score between questions and POIs. Recent years have witnessed rapid growth in various kinds of semantic similarity based QA systems such as Convolutional Neural Network Architecture (Hu et al., 2014), LSTM Based Answer Selection (Tan et al., 2015, 2016), and Cross-Attention Based Question Answering System (Hao et al., 2017). Despite the success in common landscapes, most existing studies of this family cannot work well for POI oriented QA, since it is ineffective for them to handle the unique properties of POI elements such as tags and locations. As a result, a significant gap remains between academic proposals and the industry standard of implementing location based services.

It is nontrivial to extend existing QA models to handle the challenges of POI oriented QA. In general, the unique challenges for this problem mainly

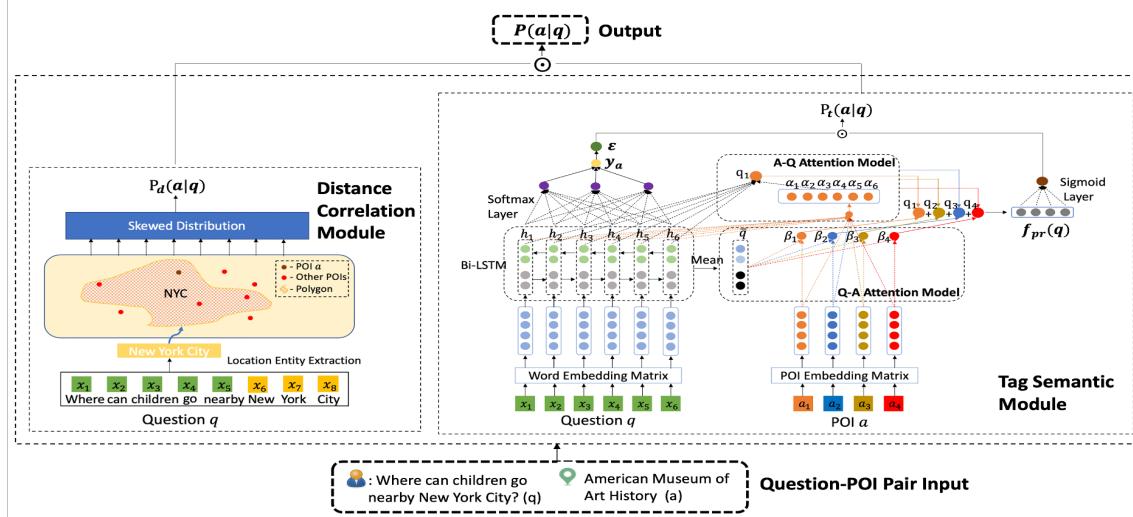


Figure 1: The overall architecture of our model. Generally, the model is made up of two parts, namely tag semantic module  $p_t$  and distance correlation module  $p_d$ . The model takes the Question-POI pair as input, and the probability of choosing a POI given the question as output.

come from two aspects. First of all, when asking POI oriented questions, people tend to emphasize certain needs, which correspond to some POI properties, such as the popular users of the POI, the service provided by the POI, and the types of the POI, etc. Hereafter we name all such POI properties as tags. Identifying such information in the question that is related to the tags of POI is crucial in this task, thus creating a bottleneck. Take the question “Where can children go nearby New York City” as an example, the word “children”, being regarded as both a question term and a POI tag, plays an important role in identifying the corresponding POIs. Second, proximity-related terms such as “nearby” and “close” deserve special treatment. Considering the same example, if there is “nearby” in the question, the candidate POIs should be mainly located outside New York City; whereas if without, the candidate POIs should be within New York City. Furthermore, for different location entities such as “nearby New York City” v.s. “nearby Manhattan”, the distance scopes of “nearby” are also different. In contrast, traditional QA methods are not able to treat these terms in their models properly and thus leading to a poor performance on POI oriented QA.

In this paper, we propose a POI oriented QA model with Joint Inference (named as PJI for short) to tackle the challenges mentioned before. PJI mainly has two modules which are named as tag semantic module and distance correlation module. The tag semantic module is used to automatically search for relevant POIs based on semantic tag in-

formation. Besides, in order to capture specific patterns buried in questions and POIs, we develop a novel cross attention based question embedding structure. Therefore the mutual influence between questions and POIs is taken into account. In the distance correlation module, we adopt a skewed distribution on three-level locations including city, district, Area of Interest (AOI) to fit the distance distribution between candidate POIs and mentioned location terms in the question. Both modules are fused together and optimized in an end-to-end manner for retrieving the final POI list. Our major contributions can be summarized as follows:

- We tackle the POI oriented QA problem by proposing a new deep learning model with joint inference.
- We leverage two neural network modules to build a bridge between questions and POIs on both POI tags and question location terms. We also adopt a skewed distribution method to deal with proximity-related terms.
- We design a special embedding structure using cross attention mechanism to obtain a more precise and flexible representation of questions.
- We conduct comprehensive experiments on two real-world datasets enabling the evaluation of the results from different perspectives. Experimental results demonstrate significant improvements of PJI over all the state-of-the-art baselines.

## 2 Related Work

**QA with Semantic Parser** Semantic parsing shines at handling complex linguistic constructions and obtains reasonable performance on question answering problems. Traditionally, semantic parsers like AMR (Banarescu et al., 2012) and SQL (Androulakakis et al., 1995) map sentences to formal representations of their underlying meaning (Shen and Lapata, 2007; Yao et al., 2014; Hill et al., 2015; Talmor et al., 2017). By leveraging a knowledge base, semantic parsing is reduced to query graph generation and stage searching.

**Neural Approaches for QA** With the recent development in deep learning, neural networks have achieved great success in question answer problems (Salakhutdinov and Hinton, 2009; Collobert et al., 2011; Socher et al., 2012; Hu et al., 2014; Tan et al., 2015, 2016). Most of these models use a deep neural network like GRU (Chung et al., 2014) and LSTM (Hochreiter and Schmidhuber, 1997) to handle the long texts required for QA. Further improvements like attention mechanism are applied to focus on the most relevant facts (Hao et al., 2017; Zhao et al., 2019). The relevance score of each QA pair is the cosine similarity of the semantic vectors. The final answers to each question are then sorted by the similarity score.

**Probabilistic Deep Learning Models** The base of probabilistic deep learning models is to use the neural network as a conditional model parameterised by the weights in the network when some inputs are given. The output is obtained by optimizing the parameters in the model with the estimates provided by Bayesian framework. Several probabilistic models have been used in tasks like question answering with knowledge graph and link prediction (Wang et al., 2007, 2014; Zhang et al., 2018). The main advantage of this complete separation of the neural network from Bayesian model is that the good features generated by the network are well used to make predictions, which gives the model high flexibility and accuracy.

## 3 Our Model

### 3.1 Preliminaries

Point of interest (POI) is a dedicated geographic entity on an online map where someone may find useful information, like a restaurant, a hotel, or a travel spot. Compared with the common entities in

knowledge graph, POI has two important properties which are tags and location. Tags refer to a short text (one or several words) in a POI describing its service (e.g. fast food or entertainment), its major users (e.g. kids or lovers), its types (e.g. restaurants or shopping mall), etc. Users are greatly facilitated by informative tags when searching for the POIs. In addition, each POI has three location properties named as location entities recording the POI located city, district, and area of interest (AOI). Here AOI refers to a polygonal area in a 2D map which usually contains several POIs, New York Central Park for example. For each location entity, it is possible to find a set of POIs within the entity.

Given a question  $q$ , the POI oriented question answering seeks to parse the question, then return a set of POIs which can be seen as the answer result according to the question. For example when  $q$  is the question “Where can children go on weekend in New York City?”, the answer is a set of POIs which are places for kids to play in New York City satisfying the information request conveyed by the user. It is possible to further rank the POIs according to some POI recommendation algorithms but it is beyond the scope of this paper.

### 3.2 Model Overview

In our framework, the dataset is a set of question-POI pairs which can be represented as  $D = \{q_i, a_i\}_{i=1}^N$ , where  $q_i$  refers to a question,  $a_i$  refers to a POI answering the question. Our model PJI aims to retrieve the correct POIs with respect to each question which corresponds to the function  $P(a_i|q_j)$  returning the probability that POI  $a_i$  satisfies the question  $q_j$ . The overall structure of our model is illustrated in Fig 1. Our model consists of two neural network modules, as described below:

**Tag Semantic Module** In the POI oriented QA, some question terms correspond to POI tags and thus serving as a bridge between questions and POIs. In Fig 1, “children” is both a term in the question and a tag of POI. However, it is not easy to match the query terms to POI tags directly. For example, terms such as “kids”, “baby” can also correspond to the tag “children”. In our model, for each question  $q_j$ , we learn the probability that a tag  $y_a$  is included in the question  $q_j$ , which can be represented as  $P(y_a|q_j)$ . Given the question embedding and tags, POIs with the corresponding tags are chosen as the answers at the tag level. We then develop a neural network module specialized for

calculating  $P(a_i|y_a, q_j)$ , which is the likelihood of POI  $a_i$  being selected given the tag  $y_a$  and the question  $q_j$ . Above all, the likelihood of choosing POI  $a_i$  as the answer to the question  $q_j$  is the marginal probability mass function over all tags:

$$p_t(a_i|q_j, \theta_t) = \sum_{y_a \in V_t} P(a_i|y_a, q_j) * P(y_a|q_j) \quad (1)$$

which sums out all possibilities of tag variables, where  $V_t$  refers to the tag set.

**Distance Correlation Module** Apart from tags, there also exists a distance correlation between questions and POIs. In this module, we first extract the location entity using NER tools, since the location entity vocabulary is very large and has fixed names. The questions usually contain some proximity-related terms, such as “nearby” and “close to”. It is hard to confidently determine whether a POI is in or out an extracted location entity polygon considering such proximity-related terms. Thus, instead of directly identifying the candidate POIs by the location entity appeared in the question, we also calculate the probability of candidate POIs considering both the distance to the extracted entity polygon and the question context. With this motivation, we introduce another probability function  $P(a_i|y_l, q_j)$ , which captures the probability of POI  $a_i$  being the answer of the question  $q_j$  if the location entity  $y_l$  appears in  $q_j$ . We denote the likelihood of choosing POI  $a_i$  given the question  $q_j$  based on distance correlation as:

$$p_d(a_i|q_j, \theta_d) = \sum_{y_l \in V_d} P(a_i|y_l, q_j) * P(y_l|q_j) \quad (2)$$

where  $P(y_l|q_j) = 1$  if  $y_l$  appears in  $q_j$ , otherwise  $P(y_l|q_j) = 0$ ,  $y_l \in V_d$  and  $V_d$  refers to the location entity set.

**Overall Formulation** With the two modules above, the parameters of the function  $p(a_i|q_j)$  can be estimated by maximizing the log-likelihood as follows:

$$\max_{\theta_t, \theta_d} \left( \frac{1}{N} \sum_{i=1}^N \log p_t + \frac{1}{N} \sum_{i=1}^N \log p_d \right) \quad (3)$$

### 3.3 Neural Network Module for Tag Semantic Matching

Due to the linguistic diversity of describing a certain tag, it is almost impossible to recognize the tag with exact matching. Therefore, we build a

tag recognizer which can be jointly trained with the model. After that, we can get the POIs given the tag and question representations with a cross attention architecture.

**QA Embedding** We use two dense  $d$  dimensional vector representations of questions in the module. The first one is represented as  $f_{ent}(\cdot) : q \rightarrow R^d$ , which takes the Word2Vec vectors as input, then feeds them into a Bi\_LSTM neural network with a pooling layer. It helps to capture the sequence information in the question and is used in POI tag recognition. The other one is denoted as  $f_{pr}(\cdot) : q \rightarrow R^d$ , which leverages attention mechanism to distinguish and catch the most important information in questions. Rather than apply a simple attention layer, we introduce a special cross attention mechanism tailored to this task originally first brought by Hao et al. (2017). The answer POI is embedded with function  $g(\cdot) : a \rightarrow R^d$ , which calculates the average value of POI tag vectors obtained from Word2Vec.

**Cross Attention Mechanism** Similar to  $f_{ent}$ , the structure  $f_{pr}$  consists of a Bi\_LSTM network with a pooling layer, whereas the output of it interacts with the POI representation and takes the attention weights into account. The final attentive embedding consists of POI-towards-question embedding and question-towards-POI embedding. In POI-towards-question step, we train weights between every state in the Bi\_LSTM hidden layer and POI tag, then get a set of weighted question vectors regarding each POI tag. The following formulas are proposed to calculate the vectors:

$$\alpha_{mn} = \text{softmax}(h(W^T[h_n; e_m] + b)) \quad (4)$$

$$f_{pr}(q)_m = \sum_n \alpha_{mn} h_n \quad (5)$$

where  $h_n$  denotes the question hidden layer vector.  $e_m$  denotes the POI tag embedding vector.  $\alpha_{mn}$  is the weight of attention from the tag  $e_m$  to the  $n$ th word in the question.  $h(\cdot)$  is an activation function.

In question-towards-POI step, we learn a set of weights between the question pooling layer vector and POI tag. Using the weighted question vectors in the first step and the weights in the second step, we can then get the final weighted double-sided attentive question vector by multiplying and adding them up.

$$f_{pr}(q) = \sum_m \beta_m f_{pr}(q)_m \quad (6)$$

$$\beta_m = \text{softmax}(h(W^T[f_{ent}(q); e_m] + b)) \quad (7)$$

where  $\beta_m$  denotes the attention of question towards answer aspects.

**POI Tag Recognition** We exploit the question context to build the tag recognizer. For instance, if the question contains the word “dating”, it means that the target audience is lovers and the POI type should be like parks and restaurants. Specifically, we embed the question to a  $d$  dimensional vector using embedding function  $f_{ent}(\cdot) : q \rightarrow R^d$  as described above. Then given the embedding vector of the question  $q$ , we set the likelihood of choosing tag  $y_a$  by adding a softmax layer as follows:

$$P(y_a|q) = \text{softmax}(W_y^T f_{ent}(q)) \quad (8)$$

$$= \frac{\exp(W_y^T f_{ent}(q))}{\sum_{y' \in V_t} \exp(W_{y'}^T f_{ent}(q))} \quad (9)$$

where  $V_t$  refers to the tag set in the POI dataset.

**Tag Based POI Retrieval** Since the number of POIs in the dataset is often very large, it is necessary to obtain some candidate POIs based on tag information and discard the irrelevant ones. Having  $P(y_a|q)$ , we can get POI tag  $y_a$  with the highest score. We then filter out POIs with the tag  $y_a$  from the dataset and form the candidate set. Precisely, we introduce a Dirac delta function  $\epsilon$  to accomplish this process. For POIs with  $y_a$ , the function  $\epsilon_{y_a}(a)$  is set to 1, while for POIs without,  $\epsilon_{y_a}(a)$  is set to 0. The filtering of POI greatly reduces the workload of subsequent process, and has a significant effect for large-scale data.

After obtaining the tag  $y_a$  in the question  $q_j$  and the function  $\epsilon$ , the next step is to retrieve the corresponding POIs, which is represented as  $P(a_i|y_a, q_j)$  in Section 3.2. Suppose questions are embedded using the embedding function  $f_{pr}(\cdot) : q \rightarrow R^d$ . In this function, the final output question embedding is the weighted cross-attentive vectors where informative patterns in questions are strongly focused. The likelihood of choosing  $a_i$  given question answer embedding and POI tag can be represented as follows:

$$P(a_i|y_a, q) = \text{sigmoid}(f_{pr}(q)^T g(a_i)) \cdot \epsilon_{y_a}(a_i) \quad (10)$$

### 3.4 Neural Network Module for Distance Correlation

This section mainly discusses the approach for matching the POIs to the question based on the aspect of distance correlation. As discussed in Section 3.2, the first step of distance correlation module is to find location entities in the question. In our model, we assume that there are three types of location entities: city, district, AOI (Area of Interest). AOI is a location entity on the map with boundaries (e.g. Central Park) which usually belongs to a district (e.g. Manhattan) of a city (e.g. New York). We first build a dictionary storing all of the location entities and their corresponding scopes as well as types. For every question, we extract the location terms in the question with an NER (named entity recognition) tool before mapping them onto the dictionary. Note that the location term extracted directly from questions can be hierarchical. For example, AOIs may appear in the form of District+AOI (e.g. Manhattan Central Park) or City+AOI (New York Central Park) or City+District+AOI (New York Manhattan Central Park) or just itself (Central Park). Thus, we set the priority order to  $\text{AOI} > \text{district} > \text{city}$  when conducting entity mapping.

**Proximity-related Terms** While retrieving the POIs according to the location entity, another factor we should consider is whether the question contains some proximity-related terms such as “nearby”, “close to”, or “neighboring”. When these terms appear in the question, people are actually expecting POIs which are close to, or outside the location boarder. It implies that the model should avoid simply returning POIs within the location polygon. Fig 2(a) shows the real-data distribution of POI with respect to questions with and without proximity-related terms according to real-world data used in our experiments.

In addition, concerning questions with proximity-related terms, the area of the location entity also has an important impact on the probability distribution of the distance between the selected POI coordinate and location entity polygon. As shown in Fig 2(b), when asking city-level questions with proximity-related terms (e.g. “Where can children go nearby New York?”), the result may contain POIs located in city suburban district or outside the city; while as for AOI-level questions (e.g. “Where can children go nearby Central Park?”), the result may only contain POIs outside

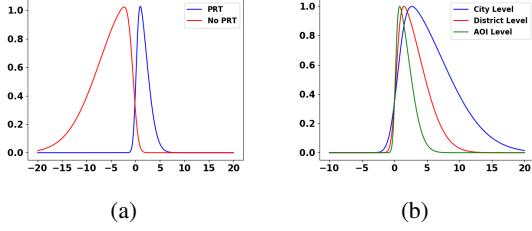


Figure 2: The POI probability distribution concerning distance. X axis is the log distance between the POI and the location entity polygon. If POI is outside the polygon, the distance is positive, otherwise is negative. Y axis is the probability the POI is recommended. (PRT denotes proximity-related terms.)

but close to the border of the AOI. This is because the area of a city is much bigger than that of an AOI. With different area sizes of the location entity, the probability distribution functions are quite different.

**Distance Correlation Calculation** The probability of choosing POI  $a_i$  given the location entity in question  $q_j$  has a proportional relationship with the distance between the POI and the location entity polygon. That is, if a POI is very far away from the expected location, the probability we recommend it is close to zero. Apart from the distance, as discussed above, proximity-related terms and location entity areas should also be taken into account when calculating the likelihood.

Given the location entity  $y_l$  extracted from the question  $q_j$ , all factors, including the distance between the polygon of  $y_l$  and POI  $a_i$ , the area size of  $y_l$  and proximity-related terms, have an impact on the likelihood distribution. Specifically, we propose a *skewed distribution* based model, which takes the distance from POI to the location entity  $d(a_i, y_l)$ , indicator function  $\tau(q_j)$ , as well as the area of location entity  $s(y_l)$  as inputs.  $\tau(q_j)$  is the indicator function that  $\tau(q_j) = -1$  if the question contains proximity-related terms, otherwise  $\tau(q_j) = 1$ . The probability of choosing POI  $a_i$  having the location entity  $y_l$  and the question  $q_j$  is:

$$P(a_i|y_l, q_j) = \text{sigmoid}(W_d f(\frac{\tau(q_j)d(a_i, y_l)}{s_y})) \quad (11)$$

$$f(x) = \frac{2}{\omega} \phi(\frac{x - \xi}{\omega}) \Phi(\alpha \frac{x - \xi}{\omega}) \quad (12)$$

$$\phi(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} \quad (13)$$

$$\Phi(x) = \int_{-\infty}^x \phi(t) dt = \frac{1}{2} (1 + \text{erf}(\frac{x}{\sqrt{2}})) \quad (14)$$

Where  $f(x)$  is the skewed normal distribution of  $x$ ,  $W_d$  is what we want to optimize. Note that  $\alpha, \xi, \omega$  are hyper parameters. Given the formulation above, we can see if the questions do not contain proximity-related terms,  $\tau(q_j)$  value is equal to 1, POIs inside the polygon scope are what we need. As for questions containing proximity-related terms, the smaller polygon area  $s_y$  is, the steeper the distribution curve will be, as a result, POIs closer to the polygon boundary will be more likely to be selected.

### 3.5 Inference

During inference, ideally we want to find the candidate POIs given the question  $q_j$ . In the aspect of POI tags, we select the tag  $y_a$  receiving the maximum score from  $P(y_a|q_j)$ . Then we reduce the candidate POI number by filtering out the POIs whose corresponding tag is equal to  $y_a$ . After that, we calculate the semantic probability of choosing the POI as the answer. In distance correlation stage, the computation is quadratic in the number of location entities and thus is too expensive. We first extract the location entity  $y_l$  by NER and calculate the distance from POI coordinates to the polygon of  $y_l$  afterwards. Take the question in Fig 1 as an example, after the extraction step, we obtain the location entity “New York City”. Finally, we select the top 5 candidate POIs with top scores as the result.

## 4 Experiments

### 4.1 Experiment Setup

**Datasets** We construct two large-scale datasets, both of which are based on queries extracted from query logs of a widely used mobile search engine App and POIs obtained from an online map service provider. In order to filter the POI related questions from the search engine App, we design a set of templates such as “where can [\*] go in [\*]” and keep all the queries that match with the templates. To construct the ground truth of question-POI pairs used in the training period, given questions satisfying the templates, we crawl the related website clicked

by the user inquiring the question, then calculate the similarity between the website text and POIs. Finally we choose POIs that are most similar to the website as the answer POI. For determining the answer POIs, we sort the POIs according to their probability and choose the top-K result as the final output. Moreover, all the datasets are anonymized due to privacy concerns.

- Dataset A. This dataset mainly contains POI related questions whose geographic entities are located in Beijing. We sample questions out of one month records satisfying the template, and construct 11,000 question-POI pairs. The question data is divided into two parts randomly. The training set contains 10,900 question-POI pairs. The testing set is made up of 100 questions which do not appear in the training set.
- Dataset B. In this dataset, the location of the questions is not restricted in Beijing. We randomly sample questions to construct the question-POI pairs which covers most of the cities and many popular visited districts and AOIs in China. Similarly, there are 350,900 question-POI pairs in the training set, and 100 questions in the testing set.

On average, the length of questions in 2 datasets is 37.8 Chinese characters. The average length of POIs including its tag information (name, tags, city, district and AOI) is 30.3 characters. We later evaluate our model on these two datasets by the percent of hits at K (%hits@K) which is the percent of question-POI pairs whose POI appears in top-K retrieved POI.

**Baselines** We compare our model with several state-of-the-art baselines to show the effectiveness of our model. The first two are semantic parser based methods using tag information and the left ones are deep learning methods based on semantic matching.

- Template Matching Method (TMM) This method first converts the questions into SQL queries according to the templates, then retrieves POIs from database.
- StanfordCoreNLP Stanford CoreNLP is an integrated NLP toolkit providing a wide range of linguistic analysis tools. We use it as a Chinese semantic parser to recognize the tags.

Based on the tool, we can turn the question into SQL queries according to the semantic characteristics of the tags.

- Bi-LSTM It is a basic deep neural network model which takes the Word2Vec vectors of query and answer as input and their cosine similarity as output (Tan et al., 2015). It utilizes a Bi-LSTM layer to capture question semantic features and then feed them into a pooling layer. This model takes the max margin hinge loss as the loss function.
- Bi-LSTM+ATT (AQA) Compared with Bi-LSTM, in this model, each Bi-LSTM output vector will be multiplied by a softmax weight, which is determined by the answer embedding.
- Bi-LSTM+C-ATT (CAQA) This is a state-of-the-art end-to-end neural question answering model introduced by (Hao et al., 2017). It considers the double-sided attention containing question-to-answer attention and answer-to-question attention.

## 4.2 Experiment Results

**Overall Performance** We compare our model with all the baselines whose results are shown in Table 1. Conclusions observed are listed as follows. (1) Compared with typical neural network based models, semantic parsing based methods have a higher %hits@K rate on the whole. However, with the lack of flexibility, their %hits@K rate is worse than our PJI model. (2) In general, models with attention mechanism reach better performance than models without. Bidirectional attention models achieve higher %hits@K rate than unidirectional one, which indicates there exists several parts in the questions as well as POI attributes that should be put emphasis on. (3) Our model achieves the best overall performance among all the models. In terms of %hits@K rate, no matter what K is, the rate of our model is beyond 95%. Our model utilizes several neural network modules instead of calculating the semantic similarity directly. Moreover, thanks to the cross-attention question embedding structure, our model puts strong emphasis on the distance and tag related patterns of both questions and POIs. In addition, we use a special probability distribution to handle questions with proximity-related geographic terms which are treated the same as normal questions in the baselines.

	Dataset A			Dataset B		
	hits@1	hits@3	hits@5	hits@1	hits@3	hits@5
TMM	84.9%	84.9%	84.9%	82.8%	82.8%	82.8%
CoreNLP	88.9%	88.9%	88.9%	86.5%	86.5%	86.5%
Bi-LSTM	55.6%	56.0%	61.7%	29.1%	29.4%	31.1%
AQA	65.2%	67.3%	68.8%	37.5%	38.9%	35.6%
CAQA	69.2%	56.1%	68.1%	42.1%	42.8%	49.0%
PJI	98.6%	98.9%	99.0%	97.2%	97.9%	99.1%

Table 1: The overall performance over two datasets.

**Three-level Location Performance.** Table 2 shows the %hits@5 of two datasets where questions contain city, district and AOI location entities, respectively. As shown in the table, no matter which model we use, city level questions obtain the best result compared to other two types. The reason is that the number of cities in the whole nation is rather small and there is almost no duplicate city names among them. However, both AOI and district names can have a lot of duplications thus causing ambiguity and noise. Moreover, due to the hierarchical nature of the location entity, AOI names appear in different formats, which increases the difficulty of POI retrieval. Therefore, the template-based method and the end-to-end similarity matching method may be far from meeting the real-world demands of POI oriented QA. Despite the challenges we mentioned above, our model still outperforms all the baselines on city, district and AOI questions.

	Dataset A			Dataset B		
	Cit.	Dis.	AOI	Cit.	Dis.	AOI
TMM	94.1%	92.0%	91.3%	93.2%	91.6%	90.2%
CoreNLP	96.0%	93.4%	58.3%	90.3%	90.1%	31.8%
Bi-LSTM	92.1%	76.2%	8.4%	90.9%	62.1%	3.5%
AQA	92.4%	78.5%	9.1%	92.2%	62.8%	9.3%
CAQA	92.9%	79.4%	9.7%	92.6%	63.1%	9.7%
PJI	100%	99.6%	97.3%	99.8%	99.2%	97.0%

Table 2: The %hits@5 rate on questions containing different location entities.

### 4.3 Proximity-related Term Analysis

Fig 3 shows the %hits@5 with and without proximity-related terms on Dataset A and B. From the result we can conclude that all existing baselines cannot handle questions with proximity-related geographic terms. For traditional neural network QA models, the model has no idea how important these words are and considers them just as normal words. As a result, the results returned do not make sense to the users. Nevertheless, this problem gets tackled by the distance probability module in our model. Therefore, our model outperforms the baselines when it comes to these kinds of problems to a great extent.

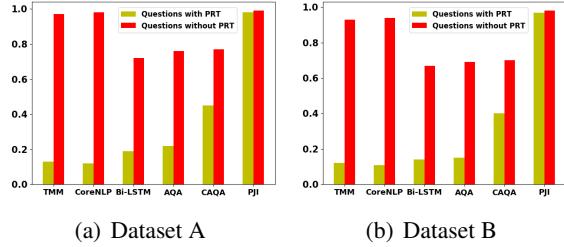


Figure 3: The %hits@5 rate concerning questions with and without proximity-related terms on two datasets.

## 5 Conclusion

In this paper, we propose a novel deep learning framework with joint inference to solve the POI oriented question answering task. Our main contributions lie in three aspects. First, this model handles the POI oriented QA with the help of tag semantic module and distance correlation module. Second, by introducing a cross attention based question embedding structure, we achieve a precise and flexible representation of questions. Third, the proposed model can overcome several challenges of POI oriented QA including POI tag recognition, proximity-related term processing and diverse distance correlation. Extensive experiments on two real-world datasets are carried out to demonstrate the effectiveness of our model. The result shows that our approach outperforms all the baselines and state-of-the-art models.

## Acknowledgments

The work described in this paper is partially supported by a grant from the Research Grant Council of the Hong Kong Special Administrative Region, China (Project Codes: 14200719).

## References

- Ion Androutsopoulos, Graeme D Ritchie, and Peter Thanisch. 1995. Natural language interfaces to databases—an introduction. *Natural language engineering*, 1(1):29–81.
- Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2012. Abstract meaning representation (amr) 1.0 specification. In *Parsing on Freebase from Question-Answer Pairs*. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Seattle: ACL, pages 1533–1544.
- Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of

- gated recurrent neural networks on sequence modeling.
- Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of machine learning research*, 12(Aug):2493–2537.
- Yanchao Hao, Yuanzhe Zhang, Kang Liu, Shizhu He, Zhanyi Liu, Hua Wu, and Jun Zhao. 2017. An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 221–231.
- Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. 2015. The goldilocks principle: Reading children’s books with explicit memory representations. *arXiv preprint arXiv:1511.02301*.
- Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.
- Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In *Advances in neural information processing systems*, pages 2042–2050.
- Ruslan Salakhutdinov and Geoffrey Hinton. 2009. Semantic hashing. *International Journal of Approximate Reasoning*, 50(7):969–978.
- Dan Shen and Mirella Lapata. 2007. Using semantic roles to improve question answering. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*, pages 12–21.
- Richard Socher, Brody Huval, Christopher D Manning, and Andrew Y Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, pages 1201–1211. Association for Computational Linguistics.
- Alon Talmor, Mor Geva, and Jonathan Berant. 2017. Evaluating semantic parsing against a simple web-based question answering model. *arXiv preprint arXiv:1707.04412*.
- Ming Tan, Cicero Dos Santos, Bing Xiang, and Bowen Zhou. 2016. Improved representation learning for question answer matching. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 464–473.
- Ming Tan, Cicero dos Santos, Bing Xiang, and Bowen Zhou. 2015. Lstm-based deep learning models for non-factoid answer selection. *arXiv preprint arXiv:1511.04108*.
- Chao Wang, Venu Satuluri, and Srinivasan Parthasarathy. 2007. Local probabilistic models for link prediction. In *Seventh IEEE international conference on data mining (ICDM 2007)*, pages 322–331. IEEE.
- Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph and text jointly embedding. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1591–1601.
- Xuchen Yao, Jonathan Berant, and Benjamin Van Durme. 2014. Freebase qa: Information extraction or semantic parsing? In *Proceedings of the ACL 2014 Workshop on Semantic Parsing*, pages 82–86.
- Yuyu Zhang, Hanjun Dai, Zornitsa Kozareva, Alexander J Smola, and Le Song. 2018. Variational reasoning for question answering with knowledge graph. In *Thirty-Second AAAI Conference on Artificial Intelligence*.
- Ji Zhao, Dan Peng, Chuhan Wu, Huan Chen, Meiyu Yu, Wanji Zheng, Li Ma, Hua Chai, Jieping Ye, and Xiaohu Qie. 2019. Incorporating semantic similarity with geographic correlation for query-poi relevance learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 1270–1277.

# Leveraging Structured Metadata for Improving Question Answering on the Web

Xinya Du<sup>1\*</sup> Adam Fourney<sup>2</sup> Robert Sim<sup>2</sup>  
Claire Cardie<sup>1</sup> Paul N. Bennett<sup>2</sup> Ahmed Hassan Awadallah<sup>2</sup>

<sup>1</sup>Department of Computer Science, Cornell University, Ithaca, NY

{xdu, cardie}@cs.cornell.edu

<sup>2</sup>Microsoft Research, Redmond, WA

{adamfo, rsim, paul.n.bennett, hassanam}@microsoft.com

## Abstract

We show that leveraging metadata information from web pages can improve the performance of models for answer passage selection/re-ranking. We propose a neural passage selection model that leverages metadata information with a fine-grained encoding strategy, which learns the representation for metadata predicates in a hierarchical way. The models are evaluated on the MS MARCO (Nguyen et al., 2016) and Recipe-MARCO datasets. Results show that our models significantly outperform baseline models, which do not incorporate metadata. We also show that the fine-grained encoding’s advantage over other strategies for encoding the metadata.

## 1 Introduction

Question answering (QA) is a long-standing task in NLP and IR. Having QA systems that perform well on real-world questions is of significant value for search engines and intelligent assistants. While some of the earliest work tackled the task of answering questions based on a large corpus (Voorhees and Tice, 2000; Voorhees, 2003; Wang et al., 2007) (albeit mostly focusing on simple fact-oriented questions), much of the recent work on QA has focused on answering questions in a less realistic setting – drawing the answer from a paragraph of text (Rajpurkar et al., 2016; Joshi et al., 2017), which is commonly referred to as machine reading comprehension (MRC).

In this work, we tackle the more realistic problem — candidate answers passages selection/re-ranking for real-world questions on the web. In contrast to both MRC and early work on QA from a large corpus, web pages often provide an additional source of knowledge. In particular, and thanks in part to the Semantic Web initiative (Berners-Lee

Textual Object	Predicate
Classic Meatloaf	/recipe/name
20 minutes	/recipe/preptime
1 hour, 10 minutes	/recipe/cooktime
1 celery rib, ...	/recipe/ingredients

Figure 1: Metadata Example from SimplyRecipes.

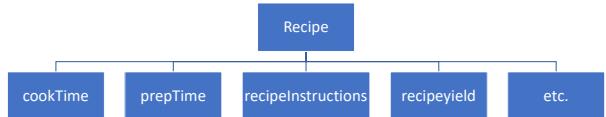


Figure 2: Hierarchy diagram showing properties of “recipe” from schema.org/recipe.

et al., 2001), it is estimated that a non-trivial portion of web pages contain metadata annotations that provide a deeper understanding of the website content. The Web Data Commons project (Mühleisen and Bizer, 2012) estimates that 0.9 billion HTML pages out of the 2.5 billion pages (37.1%) in the Common Crawl web corpus<sup>1</sup> contain structured metadata. Figure 1 shows an example of this metadata which comes in the form of object-predicate pairs annotated with schema.org tags – a set of tags/predicates defined in the schema.org<sup>2</sup> hierarchy. In the example, the hierarchical metadata is used to add more structure to the web page of a recipe, providing meaning to the otherwise unstructured content. This makes several aspects of the recipe explicit – the preparation time (PREPTIME), cooking time (COOKTIME), ingredients (INGREDIENTS), etc. Figure 2 shows the “recipe” object in schema.org; it contains several properties such as COOKTIME, PREPTIME,

\*Work conducted during internship at Microsoft Research.

<sup>1</sup><http://commoncrawl.org>

<sup>2</sup><http://schema.org>

Is selected	URL	Passage Text
x	allrecipes.com ...	Preheat oven to 350 degrees F and lightly grease a ... <span style="background-color: #f0e68c;">instructions</span>
✓	simplyrecipes.com ...	... Bake for <b>1 hour and 10 min</b> cookTime or until a meat thermometer inserted ...
x	thekitchn.com ...	Any ground meat can be used to make meatloaf: beef, pork, <b>veal</b> ingredients ...
x	livestrong.com ...	... loaf to stand for <b>10 to 15 min</b> cookTime before slicing and serving it to 4-6 yield ...

Table 1: Example of answer passage selection on the Web. There are 4 candidate passages the query “*How long should I cook ground beef meat loaf in the oven?*”

RECIPEINSTRUCTIONS, etc.

We hypothesize that leveraging this metadata, in addition to the textual content, will improve the performance of QA systems on the Web. Table 1 presents an example of a query and several candidate passages. The candidate answer passages are decorated by colored spans that denote a corresponding schema.org predicate property. The correct answer (“*1 hour and 10 min*”) could be inferred from the metadata tag COOKTIME. While it seems clear from the example that the hierarchical schema.org metadata can be exploited in web QA, it will only be of true benefit if the use of metadata is prevalent in web pages. Luckily, this is the case as shown by Guha et al. who studied a sample of 10 billion web pages and showed that one third (**31.3%**) of the pages have schema.org markup.

To date, the end-to-end web QA systems have not made use of this metadata information. We first explore how to incorporate (and the effect of incorporating) semantic web hierarchical metadata into statistical NLP models for web-based QA. More specifically, we introduce a fine-grained encoding method for metadata predicates, to better leverage the semantic information in it. We evaluate the models on the answer passage selection/re-ranking task of MS MARCO (Nguyen et al., 2016), that contains real user queries sampled from the Bing search engine, with the answer passages extracted from real-world web pages. Results show that our approaches outperform the baseline systems substantially, with more significant gains on the subset of queries whose candidate passages contain richer metadata tags. Our work demonstrates the importance of encoding metadata information for QA, and verifies our hypothesis that the metadata knowledge can significantly benefit the performance of the neural models. We also provide qualitative analysis that includes performance comparisons across

domains. Our findings further provide motivation for webmasters to annotate their web pages with semantic schema.org markup and for question answering systems developer to leverage them.

## 2 Related Work

Our work is related to several directions of work in semantic web, NLP and ML.

### Metadata for NLP and ML

Metadata like time stamp (Blei and Lafferty, 2006) and rating (Mcauliffe and Blei, 2008) have been successfully incorporated in document modeling. In community question answering, metadata is often used as hard features to improve the model performance – category metadata (Cao et al., 2010; Zhou et al., 2015) and user-level information and question- and answer-specific data (Joty et al., 2018; Xu et al., 2018). For answer quality prediction, author information (Burel et al., 2012; Suguru et al., 2016) has been often incorporated. In our work, we investigate how to leverage the *general* metadata knowledge from schema.org in web answer passage selection. Our metadata schema used, as compared to prior work mentioned, is structural and hierarchical, and applies to general web pages. The metadata could provide rich information to better understand the textual content on the web.

### Semantic Web

Berners-Lee et al. (2001) described the vision of the *Semantic Web*. The authors envisioned an extension of the World Wide Web, in which information is given well-defined meaning by bringing structure to the content of web pages. Ten years later, several major search engines have come together to launch the schema.org initiative, that to focus on creating, maintaining and promoting a common set of schemas for structured data markup on web pages. Webmasters use this schema to add metadata tags to their websites in order to help search engines understand the content. The use of such metadata has gained more popularity over the years.

## 3 Leveraging Metadata for Answer Passage Selection

In our setting of answer passage selection, the input to the system is a set of candidate passages  $p_1, \dots, p_n$ , and a query  $q$ , the goal is to identify the passage that best answers the question.

For each candidate passage  $p_i$ , we have the URL $_i$  of the web page from where it is extracted. The web document from URL $_i$ , may contain a list of metadata object-predicate pairs ( $\text{obj}_1, \text{pred}_1, \dots, (\text{obj}_m, \text{pred}_m)$ ). The detailed approach of obtaining the pairs is presented in Section (3.1). Each predicate  $\text{pred}_j$  consists of a root  $r_j$  and a property  $\text{pro}_j$  (e.g., RECIPE and COOKTIME for /RECIPE/COOKTIME, respectively). We denote the path between  $r_j$  and  $\text{pro}_j$  as  $pt_j$ .

### 3.1 Generate Metadata-Decorated Passages

Algorithm 1 generates the decorated answer passages with metadata. The example for a decorated passage is shown immediately after the algorithm. The spans are marked up with the metadata predicate features. The decorated results are later used as input for our models. To be more specific, given the queryPsgExample (including query, candidate answer passage, URL, label of whether is selected) and metadata object-predicate pairs as input, we aim to obtain the queryPsgExamples whose candidate answer passages are decorated. We first obtain all the metadata pairs (matchingMetaPairs) for the URL where the passage text appears (line 1). Then, for each metadata pair in matchingMetaPairs, we employ a similarity function (MetaSim in line 6) to first compute the similarity between all possible text spans of the passage and the object text in the metadata object-predicate pair; afterwards the function records the start and end offset of the text spans which have a similarity score higher than the threshold. In our case, we use BLEU-4 (Papineni et al., 2002) as MetaSim. It calculates a score for up to 4-grams overlap using uniform weights. A metadata-decorated candidate passage with the algorithm is presented in Table 2.

---

**Algorithm 1:** How to obtain for metadata for each URL and generate metadata-decorated passage

---

**Data:** queryPsgEg (query, psgText, URL, label), metaPairs (subj, pred, obj, URL);

```

1 matchingMetaPairs←Join(queryPsgEg[URL]==
metaPairs[URL]);
2 for each pair ∈ matchingMetaPairs do
3   if pair[obj] is not text then
4     continue;
5   else
6     startOffsets, endOffsets, score ←
      MetaSim(queryPsgEg[psgText], pair[obj]);
7     Decorate(queryPsgEg[psgText],
      startOffsets, endOffsets, pred);
8   end
9 end

```

---

word	Rinse	tilapia	fillets	in	cold	water	...
pred.	O	B_R_ING	I_R_ING	O	O	O	O
feature	O	O	O	O	B_R_ING	I_R_ING	I_R_ING

Table 2: Metadata-Decorated Candidate Passage

### 3.2 Neural Passage Selection with Fine-grained Metadata Encoding

We propose a simple but effective neural network structure for building our base neural passage selector (NPS). Similar to the neural reader (Hermann et al., 2015; Chen et al., 2017) for MRC, we first obtain a feature-rich (including the fine-grained encoding of the metadata) contextualized representation for each token in the passage and query. The output layer takes the passage and query representations as input and makes the prediction.

**Fine-grained metadata embedding** each predicate feature  $\text{pred}$  (e.g., /RECIPE/COOKTIME) includes the root  $r$  (RECIPE) and the property  $\text{pro}$  (COOKTIME). To leverage this information, we propose to leverage the hierarchy present on the predicate by learning the root embedding  $\mathbf{E}_r$ , the property embedding  $\mathbf{E}_{\text{pro}}$ , as well as the path embedding  $\mathbf{E}_{\text{pt}}$  (RECIPE→COOKTIME), instead of only learning an embedding of the entire predicate (/RECIPE/COOKTIME). Thus, the final predicate feature encoding for token  $t_i$  is the concatenation of the three components:  $\mathbf{E}_{\text{pred}}(\text{pred}_i) = \text{concat}(\mathbf{E}_r(r_i), \mathbf{E}_{\text{pro}}(\text{pro}_i), \mathbf{E}_{\text{pt}}(pt_i))$ .

**Passage & Query encoding** We first represent each token  $t_i$  in the **passage** with a vector representation and pass it through a multi-layer BiLSTM (Hochreiter and Schmidhuber, 1997) network to get the contextualized representation for each token ( $\mathbf{t}_1, \mathbf{t}_2, \dots$ ), where  $\mathbf{t}_i$  is the concatenation of:

- (*Contextualized word embedding*: GloVe 840B.300d (Pennington et al., 2014) embeddings is used to initialize the embedding layer and is fine-tuned during training, we denote it as  $\hat{\mathbf{t}}_i$  for token  $t_i$ . Besides, we also use the pretrained contextualized representations produced by BERT (Devlin et al., 2019),  $\hat{\mathbf{q}}_1, \dots, \hat{\mathbf{q}}_m, \dots, \hat{\mathbf{t}}_1, \dots, \hat{\mathbf{t}}_n = \text{BERT}([\text{CLS}], q_1, \dots, q_m, [\text{SEP}], t_1, \dots, t_n)$ . For the  $i^{th}$  token, the word embedding  $\mathbf{E}(t_i)$  is the concatenation of the two.
- *Metadata predicate embedding*: We use the fine-grained predicate encoding of metadata

pair ( $\mathbf{E}_{pred}(pred_i)$ ), as described above. Embedding for beginning (B<sub>\_</sub>) and intermediate (I<sub>\_</sub>) tokens of a decorated span are different and learned during training; For the other passage tokens that are not metadata-decorated, their predicate (O) embedding are filled with zero vectors.

- *Aligned query embedding*: Similar to (Chen et al., 2017), we also incorporate the aligned query embedding. This feature is intended to capture the similarity between  $t_i$  and each query word  $q_j$ . For the  $i^{th}$  token  $t_i$ . It is calculated as:  $\sum_j \mathbf{E}(q_j) * sim(\mathbf{E}(t_i), \mathbf{E}(q_j))$ .

The encoding  $\mathbf{p}_k$  for candidate passage  $k$  is the sum of the token representations after the BiLSTM. Similarly, **query** token embedding  $\mathbf{q}_j$  is the concatenation of its contextualized word embedding ( $\hat{\mathbf{q}}_j$ ) and the GloVe embedding. We pass it through another BiLSTM, and use the sum operation to obtain the query encoding  $\mathbf{q}$ .

**Prediction** Finally, the “Is\_selected” score for passage  $k$  is calculated as a function of the passage encoding  $\mathbf{p}_k$  and the query encoding  $\mathbf{q}$ :  $score(k) = softmax(\mathbf{p}_k W \mathbf{q})$ . At test time, we calculate  $score(1), \dots, score(n)$  for all the candidate answer passages, and select the passage with highest score:  $\text{argmax}_k(score(k))$ .

## 4 Experiments and Analysis

This section first presents the QA dataset that is used for evaluation, and then describe results comparing different methods (with or without leveraging the metadata information).

### 4.1 Datasets and Models

We evaluate our models on the passage selection task of **MS MARCO** (Nguyen et al., 2016), to our knowledge, this is currently the only large-scale real-world QA/MRC dataset on general web pages, that is paired with URLs from which the candidate passages are extracted. To measure how the models perform when trained and tested on a subset of queries from a focused domain, where the usage of schema.org metadata is more prevalent, we extract the QA pairs of the recipes domain from MS MARCO dataset and extend it with extra QA pairs in this domain (**Recipe-MARCO**). Table 3 shows the number of queries for the datasets. Although WikiQA (Yang et al., 2015) and Natural Questions (Kwiatkowski et al., 2019) also contain

	MARCO	Recipe MARCO
Train	82,326	7515
Dev	10,047	835
Test	9650	846

Table 3: Statistics of Datasets.

queries from real users, their answer candidates are restricted to be from Wikipedia. However, the adoption of schema.org tags in Wikipedia pages is very low (< 2.2%<sup>3</sup>). This is significantly less than general web pages where the adoption rate of schema.org metadata is around 31.3%. Thus we do not use these datasets for evaluation.

We follow previous work (Yang et al., 2015; Tan et al., 2018) on reporting precision@1 (P@1) and Mean Reciprocal Rank (MRR). P@1 measures whether the highest scoring answer passage returned matches the correct passage. MRR (Voorhees and Tice, 2000) evaluates the relative rank of the correct passage in the candidate passages. We compare our models to several baselines, **S-Net** (Tan et al., 2018) is a prior state-of-the-art model on MS MARCO, it also produces synthetic answers and use text generation metrics (e.g., BLEU and ROUGE-L). In this work, we only compare to its capability of passage re-ranking. **NPS** is the baseline “neural passage selector” which does not encode metadata information. It’s similar to the implementation in Dai and Callan (2019). **B-NPS** is a version of our model which builds upon NPS and *directly* encodes the entire predicate. **F-NPS** is our main model – fine-grained metadata encoding enriched neural passage selector. We also report the results of selecting the **first** and a **random** passage.

### 4.2 Results and Analysis

Table 4 shows the comparison of different methods on the candidate passage selection task. We see that: (1) By leveraging the metadata, both versions of our model (B-NPS and F-NPS) outperform the baseline NPS model; (2) With fine-grained encoding, F-NPS significantly outperforms all models in both P@1 and MRR. Particularly, F-NPS achieves higher P@1 than NPS by around 2%; (3) From the ablation study, we see the BERT pretrained representations consistently improve the performance, and leveraging the metadata information further improves it. We also present the results of different methods when trained and tested on Recipe-

<sup>3</sup><http://webdatacommons.org/structureddata/2018-12/stats/stats.html>

	MARCO		Recipe-MARCO	
	P@1	MRR	P@1	MRR
First Passage	13.89	-	15.13	-
Random	13.76	34.76	11.35	30.67
S-Net (Tan et al., 2018)	28.30	-	-	-
NPS	32.80	51.72	41.68	59.73
w/o BERT	29.57	50.10	40.24	58.39
B-NPS	33.52	52.83	43.58	61.37
F-NPS	<b>34.70*</b>	<b>54.21</b>	<b>44.37*</b>	<b>62.46</b>
w/o BERT	33.01	52.96	43.42	61.13

Table 4: Evaluation results on datasets. Statistic significance is indicated with \* ( $p < 0.05$ ).

	Prop. (%)	NPS	F-NPS
book	6.37	29.06	<b>32.81</b>
medical	13.20	30.69	<b>34.46</b>
person	11.75	29.30	<b>32.51</b>
organization	13.32	30.12	<b>33.86</b>
review	3.09	27.42	<b>35.48</b>

Table 5: Analysis of P@1 performance for models w/ and w/o metadata information in diverse domains.

MARCO. We see that the relative increase of performances for F-NPS is more substantial.

Finally, we provide analysis on both the models and the effect of encoding metadata. Since not all web pages come with metadata, we turn our attention to the results describing the model performance on the portion of queries of MS MARCO that come with *at least one* metadata item (“**M-Rich-MARCO**”). We first perform analysis to understand how often the web pages in the dataset contain markup and how it affects the models performance. We see that for each query in MS MARCO, there are around 7.9 metadata pairs for its candidate passages; and 31.6 for queries in M-Rich-MARCO. On M-Rich-MARCO, the results we get on P@1 (F-NPS: 33.13, NPS 28.79) demonstrate that the performance gap between the model that leverages the metadata is larger than the general case. This, once again, demonstrates the effect of encoding metadata knowledge.

To better understand how the models perform and the effect of metadata on specific web domains, we report in Table 5 P@1 of models (trained on *entire* MS MARCO) on domains that are richer with metadata (i.e., book, medical, person, organization and review). We observe that queries in “medical”, “person” and “organization” domains have a larger presence in the dataset ( $> 10\%$ ). The table also shows the performance of NPS and F-NPS on each domain. We see that F-NPS outperform NPS across all these domains. And the improvement is

more substantial as compared to evaluating on the entire test set (the second column of Table 4).

## 5 Conclusion

We demonstrate benefits of incorporating metadata information from web pages for improving answer passage selection model. We describe methods for obtaining metadata and decorating passages with metadata object-predicate pairs, and a fine-grained encoding strategy for leveraging metadata information in neural models. For future work, we’ll investigate metadata for other tasks such as web entity linking and extraction.

## Acknowledgments

We thank the anonymous reviewers for helpful feedback and comments.

## References

- Tim Berners-Lee, James Hendler, and Ora Lassila. 2001. The semantic web. *Scientific american*, 284(5):34–43.
- David M Blei and John D Lafferty. 2006. Dynamic topic models. In *Proceedings of the 23rd international conference on Machine learning*, pages 113–120.
- Grégoire Burel, Yulan He, and Harith Alani. 2012. Automatic identification of best answers in online enquiry communities. In *Extended Semantic Web Conference*.
- Xin Cao, Gao Cong, Bin Cui, and Christian S Jensen. 2010. A generalized framework of exploring category information for question retrieval in community question answer archives. In *Proceedings of the 19th international conference on World wide web*, pages 201–210.
- Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. *Reading Wikipedia to answer open-domain questions*. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.
- Zhuyun Dai and Jamie Callan. 2019. Deeper text understanding for ir with contextual neural language modeling. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 985–988.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. *BERT: Pre-training of deep bidirectional transformers for language understanding*. In *Proceedings of the 2019 Conference of*

*the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Minneapolis, Minnesota. Association for Computational Linguistics.

Ramanathan V Guha, Dan Brickley, and Steve Macbeth. 2016. Schema.org: evolution of structured data on the web. *Communications of the ACM*, 59(2):44–51.

Karl Moritz Hermann, Tomas Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *NeurIPS*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Shafiq Joty, Lluís Márquez, and Preslav Nakov. 2018. Joint multitask learning for community question answering using task-specific embeddings. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Jon D McAuliffe and David M Blei. 2008. Supervised topic models. In *Advances in neural information processing systems*, pages 121–128.

Hannes Mühlisen and Christian Bizer. 2012. Web data commons-extracting structured data from two large web corpora. *LDOW*, 937.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. In *Workshop on Cognitive Computation (CoCo)*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language*

*Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Sai Praneeth Suggu, Kushwanth Naga Goutham, Manoj K. Chinnakotla, and Manish Shrivastava. 2016. Hand in glove: Deep feature fusion network architectures for answer quality prediction in community question answering. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*.

Chuanqi Tan, Furu Wei, Nan Yang, Bowen Du, Weifeng Lv, and Ming Zhou. 2018. S-net: From answer extraction to answer synthesis for machine reading comprehension. In *AAAI*.

Ellen M. Voorhees. 2003. Evaluating the evaluation: A case study using the TREC 2002 question answering track. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 260–267.

Ellen M. Voorhees and Dawn M. Tice. 2000. The TREC-8 question answering track. In *Proceedings of the Second International Conference on Language Resources and Evaluation (LREC'00)*, Athens, Greece. European Language Resources Association (ELRA).

Mengqiu Wang, Noah A. Smith, and Teruko Mitamura. 2007. What is the jeopardy model? A quasi-synchronous grammar for QA. In *EMNLP-CoNLL*.

Steven Xu, Andrew Bennett, Doris Hoogeveen, Jey Han Lau, and Timothy Baldwin. 2018. Preferred answer selection in stack overflow: Better text representations... and metadata, metadata, metadata. In *Proceedings of the 2018 EMNLP Workshop WNUT: The 4th Workshop on Noisy User-generated Text*, pages 137–147.

Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. WikiQA: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2013–2018, Lisbon, Portugal. Association for Computational Linguistics.

Guangyou Zhou, Tingting He, Jun Zhao, and Po Hu. 2015. Learning continuous word embedding with metadata for question retrieval in community question answering. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 250–259, Beijing, China. Association for Computational Linguistics.

# English Intermediate-Task Training Improves Zero-Shot Cross-Lingual Transfer Too

Jason Phang<sup>1,\*</sup> Iacer Calixto<sup>1,2,\*</sup> Phu Mon Htut<sup>1</sup> Yada Pruksachatkun<sup>1</sup>  
Haokun Liu<sup>1</sup> Clara Vania<sup>1</sup> Katharina Kann<sup>3</sup> Samuel R. Bowman<sup>1</sup>

<sup>1</sup>New York University <sup>2</sup>ILLC, University of Amsterdam <sup>3</sup>University of Colorado Boulder  
[{jasonphang, iacer.calixto, bowman}@nyu.edu](mailto:{jasonphang,iacer.calixto,bowman}@nyu.edu)

## Abstract

Intermediate-task training—fine-tuning a pre-trained model on an *intermediate* task before fine-tuning again on the target task—often improves model performance substantially on language understanding tasks in monolingual English settings. We investigate whether English intermediate-task training is still helpful on *non*-English target tasks. Using nine intermediate language-understanding tasks, we evaluate intermediate-task transfer in a zero-shot cross-lingual setting on the XTReme benchmark. We see large improvements from intermediate training on the BUCC and Tatoeba sentence retrieval tasks and moderate improvements on question-answering target tasks. MNLI, SQuAD and HellaSwag achieve the best overall results as intermediate tasks, while multi-task intermediate offers small additional improvements. Using our best intermediate-task models for each target task, we obtain a 5.4 point improvement over XLM-R Large on the XTReme benchmark, setting the state of the art<sup>1</sup> as of June 2020. We also investigate continuing multilingual MLM during intermediate-task training and using machine-translated intermediate-task data, but neither consistently outperforms simply performing English intermediate-task training.

## 1 Introduction

Zero-shot cross-lingual transfer involves training a model on task data in one set of languages (or language pairs, in the case of translation) and evaluating the model on the same task in unseen languages (or pairs). In the context of natural language understanding tasks, this is generally done using a pretrained multilingual language-encoding model

\*Equal contribution.

<sup>1</sup>The state of art on XTReme at the time of final publication in September 2020 is held by Fang et al. (2020), who introduce an orthogonal method.

such as mBERT (Devlin et al., 2019a), XLM (Conneau and Lample, 2019) or XLM-R (Conneau et al., 2020) that has been pretrained with a masked language modeling (MLM) objective on large corpora of multilingual data, fine-tune it on task data in one language, and evaluate the tuned model on the same task in other languages.

Intermediate-task training (STILTs; Phang et al., 2018) consists of fine-tuning a pretrained model on a data-rich *intermediate* task, before fine-tuning a second time on the target task. Despite its simplicity, this two-phase training setup has been shown to be helpful across a range of Transformer models and target tasks (Wang et al., 2019a; Pruksachatkun et al., 2020), at least within English settings.

In this work, we propose to use intermediate training on English tasks to improve zero-shot cross-lingual transfer performance. Starting with a pretrained multilingual language encoder, we perform intermediate-task training on one or more English tasks, then fine-tune on the target task in English, and finally evaluate zero-shot on the same task in other languages.

Intermediate-task training on English data introduces a potential issue: We train the pretrained multilingual model extensively on only English data before evaluating it on non-English target task data, potentially causing the model to lose the knowledge of the other languages that was acquired during pretraining (Kirkpatrick et al., 2017; Yogatama et al., 2019). To mitigate this issue, we experiment with mixing in multilingual MLM training updates during the intermediate-task training. In the same vein, we also conduct a case study where we machine-translate intermediate task data from English into three other languages (German, Russian and Swahili) to investigate whether intermediate training on these languages improves target task performance in the same languages.

Concretely, we use the pretrained XLM-R (Con-

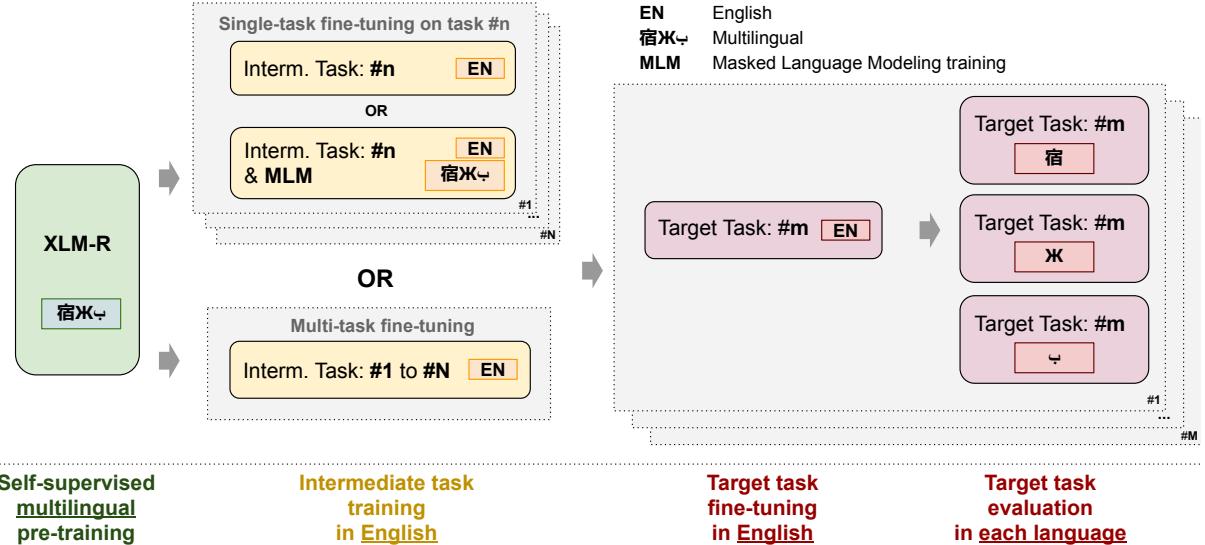


Figure 1: We investigate the benefit of injecting an additional phase of intermediate-task training on English language task data. We also consider variants using multi-task intermediate-task training, as well as continuing multilingual MLM during intermediate-task training. Best viewed in color.

neau et al., 2020) encoder and perform experiments on 9 target tasks from the recently introduced XTReme benchmark (Hu et al., 2020), which aims to evaluate zero-shot cross-lingual transfer performance across diverse target tasks across up to 40 languages each. We investigate how training on 9 different intermediate tasks, including question answering, sentence tagging, sentence completion, paraphrase detection, and natural language inference impacts zero-shot cross-lingual transfer performance. We find the following:

- Intermediate-task training on SQuAD, MNLI, and HellaSwag yields large target-task improvements of 8.2, 7.5, and 7.0 points on the development set, respectively. Multi-task intermediate-task training on all 9 tasks performs best, improving by 8.7 points.
- Applying intermediate-task training to BUCC and Tatoeba, the two sentence retrieval target tasks that have no training data of their own, yields dramatic improvements with almost every intermediate training configuration. TyDiQA shows consistent improvements with many intermediate tasks, whereas XNLI does not see benefits from intermediate training.
- Evaluating our best performing models for each target task on the XTReme benchmark yields an average improvement of **5.4 points**, setting the state of the art as of writing.

- Training on English intermediate tasks outperforms the more complex alternatives of (i) continuing multilingual MLM during intermediate-task training, and (ii) using machine-translated intermediate-task data.

## 2 Approach

We follow a three-phase approach to training, illustrated in Figure 1: (i) we use a publicly available model pretrained on raw multilingual text using MLM; (ii) we perform intermediate-task training on one or more English intermediate tasks; and (iii) we fine-tune the model on English target-task training data, before evaluating it on target-task test data in each target language.

In phase (ii), our intermediate tasks have English input data. In Section 2.4, we investigate an alternative where we machine-translate intermediate-task data to other languages, which we use for training. We experiment with both single- and multi-task training for intermediate-task training. We use target tasks from the recent XTReme benchmark for zero-shot cross-lingual transfer.

### 2.1 Intermediate Tasks

We study the effect of intermediate-task training (STILTs; Phang et al., 2018) with nine different English intermediate tasks, described in Table 1.

We choose the tasks below based to cover a variety of task formats (classification, question answering, and multiple choice) and based on evidence

Name	Train	Dev	Test	Task	Genre/Source
<b>Intermediate tasks</b>					
ANLI <sup>+</sup>	1,104,934	22,857	—	natural language inference	Misc.
MNLI	392,702	20,000	—	natural language inference	Misc.
QQP	363,846	40,430	—	paraphrase detection	Quora questions
SQuAD v2.0	130,319	11,873	—	span extraction	Wikipedia
SQuAD v1.1	87,599	10,570	—	span extraction	Wikipedia
HellaSwag	39,905	10,042	—	sentence completion	Video captions & Wikihow
CCG	38,015	5,484	—	tagging	Wall Street Journal
Cosmos QA	25,588	3,000	—	question answering	Blogs
CommonsenseQA	9,741	1,221	—	question answering	Crowdsourced responses
<b>Target tasks (XTREME Benchmark)</b>					
XNLI	392,702	2,490	5,010	natural language inference	Misc.
PAWS-X	49,401	2,000	2,000	paraphrase detection	Wiki/Quora
POS	21,253	3,974	47–20,436	tagging	Misc.
NER	20,000	10,000	1,000–10,000	named entity recognition	Wikipedia
XQuAD	87,599	34,726	1,190	question answering	Wikipedia
MLQA	87,599	34,726	4,517–11,590	question answering	Wikipedia
TyDiQA-GoldP	3,696	634	323–2,719	question answering	Wikipedia
BUCC	—	—	1,896–14,330	sentence retrieval	Wiki / news
Tatoeba	—	—	1,000	sentence retrieval	Misc.

Table 1: Overview of the intermediate tasks (top) and target tasks (bottom) in our experiments. For target tasks, *Train* and *Dev* correspond to the English training and development sets, while *Test* shows the range of sizes for the target-language test sets for each task. XQuAD, TyDiQA and Tateoba do not have separate held-out development sets.

of positive transfer from literature. [Pruksachatkun et al. \(2020\)](#) shows that MNLI (of which ANLI<sup>+</sup> is a superset), CommonsenseQA, Cosmos QA and HellaSwag yield positive transfer to a range of downstream English-language tasks in intermediate training. CCG involves token-wise prediction and is similar to the POS and NER target tasks. Both versions of SQuAD are widely-used question-answering tasks, while QQP is semantically similar to sentence retrieval target tasks (BUCC and Tateoba) as well as PAWS-X, another paraphrase-detection task.

**ANLI + MNLI + SNLI (ANLI<sup>+</sup>)** The Adversarial Natural Language Inference dataset ([Nie et al., 2020](#)) is collected using model-in-the-loop crowdsourcing as an extension of the Stanford Natural Language Inference (SNLI; [Bowman et al., 2015](#)) and Multi-Genre Natural Language Inference (MNLI; [Williams et al., 2018](#)) corpora. We follow [Nie et al. \(2020\)](#) and use the concatenated ANLI, MNLI and SNLI training sets, which we refer to as ANLI<sup>+</sup>. For all three natural language inference tasks, examples consist of premise and hypothesis sentence pairs, and the task is to classify the relationship between the premise and hypothesis as entailment, contradiction, or neutral.

**CCG** CCGbank ([Hockenmaier and Steedman, 2007](#)) is a conversion of the Penn Treebank into Combinatory Categorial Grammar (CCG) derivations. The CCG supertagging task that we use consists of assigning lexical categories to individual word tokens, which together roughly determine a full parse.<sup>2</sup>

**CommonsenseQA** CommonsenseQA ([Talmor et al., 2019](#)) is a multiple-choice QA dataset generated by crowdworkers based on clusters of concepts from ConceptNet ([Speer et al., 2017](#)).

**Cosmos QA** Cosmos QA is multiple-choice commonsense-based *reading comprehension* dataset ([Huang et al., 2019b](#)) generated by crowdworkers, with a focus on the causes and effects of events.

**HellaSwag** HellaSwag ([Zellers et al., 2019](#)) is a commonsense reasoning dataset framed as a four-way multiple choice task, where examples consist of an incomplete paragraph and four choices of spans, only one of which is a plausible continuation of the scenario. It is built using adversarial filtering ([Zellers et al., 2018; Le Bras et al., 2020](#)) with BERT.

<sup>2</sup>If a word is tokenized into sub-word tokens, we use the representation of the first token for the tag prediction for that word as in [Devlin et al. \(2019a\)](#).

**MNLI** In addition to the full ANLI<sup>+</sup>, we also consider the MNLI task as a standalone intermediate task because of its already large and diverse training set.

**QQP** Quora Question Pairs<sup>3</sup> is a paraphrase detection dataset. Examples in the dataset consist of two questions, labeled for whether they are semantically equivalent.

**SQuAD** Stanford Question Answering Dataset (Rajpurkar et al., 2016, 2018) is a question-answering dataset consisting of passages extracted from Wikipedia articles and crowd-sourced questions and answers. In SQuAD version 1.1, each example consists of a context passage and a question, and the answer is a text span from the context. SQuAD version 2.0 includes additional questions with no answers, written adversarially by crowd-workers. We use both versions in our experiments.

## 2.2 Target Tasks

We use the 9 target tasks from the XTREME benchmark, which span 40 different languages (hereafter referred to as the *target languages*): Cross-lingual Question Answering (**XQuAD**; Artetxe et al., 2020b); Multilingual Question Answering (**MLQA**; Lewis et al., 2020); Typologically Diverse Question Answering (**TyDiQA-GoldP**; Clark et al., 2020); Cross-lingual Natural Language Inference (**XNLI**; Conneau et al., 2018); Cross-lingual Paraphrase Adversaries from Word Scrambling (**PAWS-X**; Yang et al., 2019); Universal Dependencies v2.5 (Nivre et al., 2018) POS tagging; Wikiann NER (Pan et al., 2017); BUCC (Zweigebaum et al., 2017, 2018), which requires identifying parallel sentences from corpora of different languages; and **Tatoeba** (Artetxe and Schwenk, 2019), which involves aligning pairs of sentences with the same meaning.

Among the 9 tasks, BUCC and Tatoeba are sentence retrieval tasks that do not include training sets, and are scored based on the similarity of learned representations (see Appendix A). XQuAD, TyDiQA and Tatoeba do not include development sets separate from the test sets.<sup>4</sup> For all XTREME tasks, we follow the training and evaluation protocol described in the benchmark paper (Hu et al., 2020)

and their sample implementation.<sup>5</sup> Intermediate- and target-task statistics are shown in Table 1.

## 2.3 Multilingual Masked Language Modeling

Our setup requires that we train the pretrained multilingual model extensively on English data before using it on a non-English target task, which can lead to the catastrophic forgetting of other languages acquired during pretraining. We investigate whether continuing to train on the multilingual MLM pretraining objective while fine-tuning on an English intermediate task can prevent catastrophic forgetting of the target languages and improve downstream transfer performance.

We construct a multilingual corpus across the 40 languages covered by the XTREME benchmark using Wikipedia dumps from April 14, 2020 for each language and the MLM data creation scripts from the `jiant` 1.3 library (Phang et al., 2020). In total, we use 2 million sentences sampled across all 40 languages using the sampling ratio from Conneau and Lample (2019) with  $\alpha = 0.3$ .

## 2.4 Translated Intermediate-Task Training

Large-scale labeled datasets are rarely available in languages other than English for most language-understanding benchmark tasks. Given the availability of increasingly performant machine translation models, we investigate if using machine-translated intermediate-task data can improve same-language transfer performance, compared to using English intermediate task data.

We translate training and validation data of three intermediate tasks: QQP, HellaSwag, and MNLI. We choose these tasks based on the size of the training sets and because their example-level (rather than word-level) labels can be easily mapped onto translated data. To translate QQP and HellaSwag, we use pretrained machine translation models from OPUS-MT (Tiedemann and Thottingal, 2020). These models are trained with Marian-NMT (Junczys-Dowmunt et al., 2018) on OPUS data (Tiedemann, 2012), which integrates several resources depending on the available corpora for the language pair. For MNLI, we use the publicly available machine-translated training data of XNLI provided by the XNLI authors.<sup>6</sup> We use German, Russian, and Swahili translations of

<sup>3</sup><http://data.quora.com/First-Quora-DatasetRelease-Question-Pairs>

<sup>4</sup>UDPOS also does not include development sets for Kazakh, Thai, Tagalog or Yoruba.

<sup>5</sup><https://github.com/google-research/xtreme>

<sup>6</sup>According to Conneau et al. (2018), these data are translated using a Facebook internal machine translation system.

all three datasets instead of English data for the intermediate-task training.

### 3 Experiments and Results

#### 3.1 Models

We use the pretrained XLM-R Large model (Conneau et al., 2020) as a starting point for all our experiments, as it currently achieves state-of-the-art performance on many zero-shot cross-lingual transfer tasks.<sup>7</sup> Details on intermediate- and target-task training can be found in Appendix A.

**XLM-R** For our baseline, we directly fine-tune the pretrained XLM-R model on each target task’s English training data (if available) and evaluate zero-shot on non-English data, closely following the sample implementation for the XTREME benchmark.

**XLM-R + Intermediate Task** In our main approach, as described in Figure 1, we include an additional intermediate-task training phase before training and evaluating on the target tasks as described above.

We also experiment with multi-task training on all available intermediate tasks. We follow Rafel et al. (2020) and sample batches of examples for each task with probability  $r_m = \frac{\min(e_m, K)}{\sum(\min(e_m, K))}$ , where  $e_m$  is the number of examples in task  $m$  and the constant  $K = 2^{17}$  limits the oversampling of data-rich tasks.

**XLM-R + Intermediate Task + MLM** To incorporate multilingual MLM into the intermediate-task training, we treat multilingual MLM as an additional task for intermediate training, using the same multi-task sampling strategy as above.

**XLM-R + Translated Intermediate Task** We translate intermediate-task training and validation data for three tasks and fine-tune XLM-R on translated intermediate-task data before we train and evaluate on the target tasks.

#### 3.2 Software

Experiments were carried out using the jiant (Phang et al., 2020) library (2.0 alpha), based on PyTorch (Paszke et al., 2019) and Transformers (Wolf et al., 2019).

<sup>7</sup>XLM-R Large (Conneau et al., 2020) is a 550m-parameter variant of the RoBERTa masked language model (Liu et al., 2019b) trained on a cleaned version of CommonCrawl on 100 languages. Notably, Yoruba is used in the POS and NER XTREME tasks but not in the set of 100 languages.

### 3.3 Results

We train three versions of each intermediate-task model with different random seeds. For each run, we compute the average target-task performance across languages, and report the median performance across the three random seeds.

**Intermediate-Task Training** As shown in Table 2, no single intermediate task yields positive transfer across all target tasks. The target tasks TyDiQA, BUCC and Tatoeba see consistent gains from most or all intermediate tasks. In particular, BUCC and Tatoeba, the two sentence retrieval tasks with no training data, benefit universally from intermediate-task training. PAWS-X, NER, XQuAD and MLQA also exhibit gains with the additional intermediate-task training on some intermediate tasks. On the other hand, we find generally no or negative transfer to XNLI and POS.

Among the intermediate tasks, we find that MNLI performs best; with meaningful improvements across the PAWS-X, TyDiQA, BUCC and Tatoeba tasks. ANLI<sup>+</sup>, SQuAD v1.1, SQuAD v2.0 and HellaSwag also show strong positive transfer performance: SQuAD v1.1 shows strong positive transfer across all three QA tasks, SQuAD v2.0 shows the most positive transfer to TyDiQA, while HellaSwag shows the most positive transfer to NER and BUCC tasks. ANLI<sup>+</sup> does not show any improvement over MNLI (of which it is a superset), even on XNLI for which it offers additional directly relevant training data. This mirrors negative findings from Nie et al. (2020) on NLI evaluations and Bowman et al. (2020) on transfer within English. QQP significantly improves sentence retrieval-task performance, but has broadly negative transfer to the other target tasks.<sup>8</sup> CCG also has relatively poor transfer performance, consistent with Pruk-sachatkun et al. (2020).

Among our intermediate tasks, both SQuAD v1.1 and MNLI also serve as training sets for target tasks (for XNLI and XQuAD/MLQA respectively). While both tasks show overall positive transfer, SQuAD v1.1 actually markedly improves the performance in XQuAD and MLQA, while MNLI slightly hurts XNLI performance. We hypothesize that the somewhat surprising improvements to XQuAD and MLQA performance from SQuAD v1.1 arise due to the baseline XQuAD and MLQA

<sup>8</sup>For QQP, on 2 of the 3 random seeds the NER model performed extremely poorly, leading to the large negative transfer of -45.4.

	Target tasks										
Metric # langs.	XNLI acc. 15	PAWS-X acc. 7	POS F1 33	NER F1 40	XQuAD F1 / EM 11	MLQA F1 / EM 7	TyDiQA F1 / EM 9	BUCC F1 5	Tatoeba acc. 37	Avg. — —	
Without MLM	<b>XLM-R</b>	<b>80.1</b>	86.5	75.7	62.8	76.1 / 60.0	70.1 / 51.5	65.6 / 48.2	71.5	31.0	67.2
	ANLI <sup>+</sup>	- 0.8	- 0.0	- 1.4	- 3.5	- 1.1 / - 0.5	- 0.6 / - 0.8	- 0.6 / - 3.0	+19.9	+48.2	+ 6.6
	MNLI	- 1.2	+ 1.4	- 0.7	+ 0.5	- 0.3 / - 0.1	+ 0.2 / + 0.2	- 1.0 / - 1.6	+20.0	+48.8	+ 7.5
	QQP	- 4.4	- 4.8	- 6.5	- 45.4	- 3.8 / - 3.8	- 3.9 / - 4.4	-11.1 / -10.2	+17.1	+49.5	- 1.5
	SQuADv1.1	- 1.9	+ 1.2	- 0.8	- 0.4	+ 1.8 / + 2.5	+ 2.2 / + 2.6	+ 9.7 / +10.8	+18.9	+41.3	+ 8.1
	SQuADv2	- 1.6	+ 1.9	- 1.1	+ 0.8	- 0.5 / + 0.7	- 0.4 / + 0.1	+10.4 / +11.3	+19.3	+43.4	+ 8.2
	HellaSwag	- 7.1	+ 1.8	- 0.7	+ 1.6	- 0.0 / + 0.5	- 0.1 / + 0.2	- 0.0 / - 1.0	+20.3	+47.6	+ 7.0
	CCG	- 2.6	- 3.4	- 2.0	- 1.5	- 1.5 / - 1.3	- 1.6 / - 1.5	- 2.8 / - 6.2	+11.7	+41.9	+ 4.1
	CosmosQA	- 2.1	- 0.3	- 1.4	- 1.5	- 0.9 / - 1.3	- 1.5 / - 2.0	+ 0.5 / - 0.6	+19.2	+43.9	+ 6.1
	CSQA	- 2.9	- 2.8	- 1.7	- 1.6	- 1.0 / - 1.8	- 1.0 / - 0.6	+ 3.5 / + 2.9	+18.1	+48.6	+ 6.5
With MLM	Multi-task	- 0.9	+ 1.7	- 1.0	+ 1.8	+ 0.3 / + 0.9	+ 0.2 / + 0.5	+ 5.8 / + 6.0	+19.6	+49.9	+ 8.7
	ANLI <sup>+</sup>	- 1.1	+ 1.4	+ 0.0	+ 0.4	- 1.9 / - 1.7	- 0.7 / - 0.6	+ 0.9 / + 0.5	+18.6	+46.2	+ 7.1
	MNLI	- 0.7	+ 1.6	- 1.6	+ 1.0	- 0.7 / + 0.1	+ 0.4 / + 0.8	- 1.8 / - 3.2	+17.1	+44.3	+ 6.6
	QQP	- 1.3	- 1.1	- 2.4	- 0.9	- 0.3 / - 0.2	+ 0.0 / + 0.2	- 1.6 / - 4.2	+14.4	+39.8	+ 5.0
	SQuADv1.1	- 2.6	+ 0.3	- 2.0	- 0.9	+ 0.2 / + 1.6	+ 0.1 / + 1.1	+ 8.5 / + 9.5	+16.0	+40.3	+ 6.8
	SQuADv2	- 1.7	+ 2.1	- 1.4	+ 1.0	- 0.8 / + 0.1	- 0.8 / - 0.5	+ 8.3 / + 8.9	+15.6	+31.3	+ 6.1
	HellaSwag	- 3.3	+ 2.0	- 0.7	+ 0.8	- 0.8 / - 0.0	+ 0.1 / + 0.6	+ 0.3 / + 1.0	+ 6.3	+22.3	+ 3.1
	CCG	- 1.0	- 1.3	- 1.2	- 1.9	- 1.9 / - 2.2	- 2.1 / - 2.6	- 5.5 / - 6.2	+ 8.8	+36.1	+ 3.3
	CosmosQA	- 1.0	- 1.0	- 1.6	- 3.8	- 3.1 / - 3.3	- 3.7 / - 4.2	- 0.6 / - 3.2	+15.5	+42.7	+ 4.7
	CSQA	- 0.5	+ 0.3	- 1.0	- 0.7	- 0.9 / - 1.0	- 0.7 / - 0.6	+ 2.1 / + 0.4	+11.6	+17.2	+ 2.9
XTREME Benchmark Scores <sup>†</sup>											
XLM-R (Hu et al., 2020)	79.2	86.4	72.6	<b>65.4</b>	76.6 / 60.8	71.6 / 53.2	65.1 / 45.0	66.0	57.3	68.1	
XLM-R (Ours)	79.5	86.2	74.0	62.6	76.1 / 60.0	70.2 / 51.2	65.6 / 48.2	64.5	31.0	64.8	
Our Best Models <sup>‡</sup>	<b>80.0</b>	<b>87.9</b>	<b>74.4</b>	64.0	<b>78.7 / 63.3</b>	<b>72.4 / 53.7</b>	<b>76.0 / 59.5</b>	<b>71.9</b>	<b>81.2</b>	<b>73.5</b>	
Human (Hu et al., 2020)	92.8	97.5	97.0	-	91.2 / 82.3	91.2 / 82.3	90.1 / -	-	-	-	

Table 2: Intermediate-task training results. We compute the average target task performance across all languages, and report the median over 3 separate runs with different random seeds. Multi-task experiments use all intermediate tasks. We underline the best results per target task with and without intermediate MLM co-training, and bold-face the best overall scores for each target task. <sup>†</sup>: XQuAD, TyDiQA and Tatoeba do not have held-out test data and are scored using development sets in the benchmark. <sup>‡</sup>: Results obtained with our best-performing intermediate task configuration for each target task, selected based on the development set. The results for individual languages can be found in Appendix B.

models being under-trained. For all target-task fine-tuning, we follow the sample implementation for target task training in the XTREME benchmark, which trains on SQuAD for only 2 epochs. This may explain why an additional phase of SQuAD training can improve performance. Conversely, the MNLI-to-XNLI model might be over-trained, given the MNLI training set is approximately 4 times as large as the SQuAD v1.1 training set.

**Multi-Task Training** Multi-task training on all intermediate tasks attains the best overall average performance on the XTREME tasks, and has the most positive transfer to NER and Tatoeba tasks. However, the overall margin of improvement over the best single intermediate-task model is relatively small (only 0.3, over MNLI), while requiring significantly more training resources. Many single intermediate-task models also outperform the multi-task model in individual target tasks. Wang et al. (2019b) also found more mixed results from a having an initial phase of multi-task training, albeit

only among English language tasks across a different set of tasks. On the other hand, multi-task training precludes the need to do intermediate-task model selection, and is a useful method for incorporating multiple, diverse intermediate tasks.

**MLM** Incorporating MLM during intermediate-task training shows no clear trend. It reduces negative transfer, as seen in the cases of CommonsenseQA and QQP, but it also tends to somewhat reduce positive transfer. The reductions in positive transfer are particularly significant for the BUCC and Tatoeba tasks, although the impact on TyDiQA is more mixed. On balance, we do not see that incorporating MLM improves transfer performance.

**XTREME Benchmark Results** At the bottom of Table 2, we show results obtained by XLM-R on the XTREME benchmark as reported by Hu et al. (2020), results obtained with our re-implementation of XLM-R (i.e. our baseline), and results obtained with our best models, which use intermediate-task configuration selected according

TL	Model	XNLI	PAWS-X	POS	NER	XQuAD	MLQA	TyDiQA	BUCC	Tatoeba
English	<b>XLM-R</b>	<b>89.3</b>	93.4	95.9	81.6	<b>86.3 / 74.2</b>	81.6 / 68.6	70.4 / 56.6	–	–
	MNLI <sub>en</sub>	-1.2	+1.6	+0.3	+2.6	-2.1 / -1.6	+1.1 / +1.4	+1.1 / +1.1	–	–
	QQP <sub>en</sub>	-3.2	-0.4	-2.2	-5.8	-4.0 / -3.6	-2.6 / -2.6	-6.2 / -5.0	–	–
	HellaSwag <sub>en</sub>	-0.8	+1.5	+0.6	+2.7	-0.2 / +1.4	+1.8 / +2.3	+1.7 / +2.5	–	–
German	<b>XLM-R</b>	<b>83.8</b>	88.1	88.6	78.6	77.7 / 61.2	<b>69.1 / 52.0</b>	–	77.7	63.9
	MNLI <sub>en</sub>	-0.8	+0.9	-0.1	-0.8	-0.3 / -1.0	-1.0 / -0.2	–	+16.5	+32.7
	MNLI <sub>de</sub>	-0.4	+0.5	-0.3	-0.9	+0.2 / -0.3	-2.4 / -2.0	–	+17.0	+33.7
	QQP <sub>en</sub>	-2.2	-4.2	-3.2	-7.3	-4.5 / -4.7	-6.7 / -6.4	–	+16.5	+32.6
	QQP <sub>de</sub>	-2.6	-9.1	-3.2	-22.9	-6.6 / -5.9	-7.7 / -6.6	–	+16.0	+33.5
	HellaSwag <sub>en</sub>	-0.3	+0.3	+0.1	+0.5	+1.0 / +0.2	-0.3 / +0.4	–	+16.9	+33.8
Russian	HellaSwag <sub>de</sub>	-0.2	+0.2	-0.4	-0.4	+0.2 / -0.2	-3.5 / -2.5	–	+16.3	+33.5
	<b>XLM-R</b>	79.2	–	<b>89.5</b>	69.3	77.7 / 59.8	–	65.4 / 43.6	79.2	42.1
	MNLI <sub>en</sub>	+0.3	–	-0.0	+0.8	+0.1 / +1.5	–	-1.5 / -4.6	+14.3	+47.1
	MNLI <sub>ru</sub>	-0.6	–	-0.3	+1.9	-0.4 / +1.3	–	+11.2 / +16.1	+13.1	+48.3
	QQP <sub>en</sub>	-0.7	–	-2.9	-18.6	-3.5 / -2.4	–	-8.1 / -5.4	+14.1	+49.5
	QQP <sub>ru</sub>	-3.0	–	-10.6	-59.1	-5.2 / -3.9	–	-14.4 / -12.1	+13.3	+46.7
Swahili	HellaSwag <sub>en</sub>	-0.9	–	-0.0	+1.4	+0.8 / +2.9	–	-4.0 / -10.6	+14.7	+49.9
	HellaSwag <sub>ru</sub>	-0.3	–	-0.4	+2.8	+0.2 / +0.2	–	+8.5 / +13.2	-71.6	-23.5
	<b>XLM-R</b>	<b>72.4</b>	–	–	69.8	–	–	67.2 / 48.7	–	7.9
	MNLI <sub>en</sub>	-3.0	–	–	+0.6	–	–	-0.3 / -0.2	–	+24.9
	MNLI <sub>sw</sub>	-1.1	–	–	-2.4	–	–	+13.8 / +23.4	–	+47.9
	QQP <sub>en</sub>	-2.8	–	–	-4.6	–	–	-12.7 / -12.2	–	+27.2
	QQP <sub>sw</sub>	-7.1	–	–	-32.1	–	–	-7.0 / -0.4	–	+41.8
	HellaSwag <sub>en</sub>	-0.4	–	–	+0.1	–	–	-0.9 / -0.4	–	+27.2
	HellaSwag <sub>sw</sub>	-9.8	–	–	+0.4	–	–	+15.6 / +26.3	–	-0.5

Table 3: Experiments with translated intermediate-task training and validation data evaluated on all XTREME target tasks. In each target language (TL) block, models are evaluated on a single target language. We show results for models trained on original intermediate-task training data (*en*) and compare it to models trained on translated data {*de*, *ru*, *sw*}. ‘–’ indicates that target task data is not available for that target language.

to development set performance on each target task. Based on the results in Table 2, which reflect the median over 3 runs, we pick the best intermediate-task configuration for each target task, and then choose the best model out of the 3 runs. Scores on the XTREME benchmark are computed based on the respective test sets where available, and based on development sets for target tasks without separate held-out test sets. We are generally able to replicate the best reported XLM-R baseline results, except for Tatoeba, where our implementation significantly underperforms the reported scores in Hu et al. (2020), and TyDiQA, where our implementation outperforms the reported scores. We also highlight that there is a large margin of difference between development and test set scores for BUCC—this is likely because BUCC is evaluated based on sentence retrieval over the given set of input sentences, and the test sets for BUCC are generally much larger than the development sets.

Our best models show gains in 8 out of the 9 XTREME tasks relative to both baseline implementations, attaining an average score of 73.5 across target tasks, a 5.4 point improvement over the pre-

vious best reported average score of 68.1. We set the state of the art on the XTREME benchmark as of June 2020, though Fang et al. (2020) achieve higher results and hold the state of the art using an orthogonal approach at the time of our final publication in September 2020.

### Translated Intermediate-Task Training Data

In Table 3, we show results for experiments using machine-translated intermediate-training data, and evaluated on the available target-task languages. Surprisingly, even when evaluating in-language, using target-language intermediate-task data does not consistently outperform using English intermediate-task data in any of the intermediate tasks on average.

In general, cross-lingual transfer to XNLI is negative regardless of the intermediate-task or the target language. In contrast, we observe mostly positive transfer on BUCC, and Tatoeba, with a few notable exceptions where models fail catastrophically. TyDiQA exhibits positive transfer where the intermediate- and target-task languages aligned: intermediate training on Russian or German helps TyDiQA performance in that respective language,

whereas intermediate training on English hurts non-English performance somewhat. For the remaining tasks, there appears to be little correlation between performance and the alignment of intermediate- and target-task languages. English language QQP already has mostly negative transfer to all target tasks except for BUCC and Tatoeba (see Table 2), and also shows a similar trend when translated into any of the three target languages.

We note that the quality of translations may affect the transfer performance. While validation performance on the translated intermediate tasks (Table 15) for MNLI and QQP is only slightly worse than the original English versions, the performance for the Russian and Swahili HellaSwag is much worse and close to chance. Despite this, intermediate-task training on Russian and Swahili HellaSwag improve performance on PAN-X and TyDiQA, while we see generally poor transfer performance from QQP. The interaction between translated intermediate-task data and transfer performance continues to be a complex open question. Artetxe et al. (2020a) found that translating or back-translating training data for a task can improve zero-shot cross-lingual performance for tasks such as XNLI depending on how the multilingual datasets are created. In contrast, we train on translated intermediate-task data and then fine-tune on a target task with English training data (excluding BUCC2018 and Tatoeba). The authors of the XTREME benchmark have also recently released translated versions of all the XTREME task training data, which we hope will prompt further investigation into this matter.

## 4 Related work

Sequential transfer learning using pretrained Transformer-based encoders (Phang et al., 2018) has been shown to be effective for many text classification tasks. This setup generally involves fine-tuning on a single task (Pruksachatkun et al., 2020; Vu et al., 2020) or multiple tasks (Liu et al., 2019a; Wang et al., 2019b; Raffel et al., 2020), sometimes referred to as the intermediate task(s), before fine-tuning on the target task. We build upon this line of work, focusing on intermediate-task training for improving cross-lingual transfer.

Early work on cross-lingual transfer mostly relies on the availability of parallel data, where one can perform translation (Mayhew et al., 2017) or project annotations from one language into another

(Hwa et al., 2005; Agić et al., 2016). For dependency parsing, McDonald et al. (2011) use delexicalized parsers trained on source languages and labeled training data for parsing target-language data. Agić (2017) proposes a parser selection method to select the single best parser for a target language.

For large-scale cross-lingual transfer outside NLU, Johnson et al. (2017) train a single multilingual neural machine translation system with up to 7 languages and perform zero-shot translation without explicit bridging between the source and target languages. Aharoni et al. (2019) expand this approach to cover over 100 languages in a single model. Recent works on extending pretrained Transformer-based encoders to multilingual settings show that these models are effective for cross-lingual tasks and competitive with strong monolingual models on the XNLI benchmark (Devlin et al., 2019b; Conneau and Lample, 2019; Conneau et al., 2020; Huang et al., 2019a). More recently, Artetxe et al. (2020a) showed that cross-lingual transfer performance can be sensitive to translation artifacts arising from a multilingual datasets’ creation procedure.

Finally, Pfeiffer et al. (2020) propose adapter modules that learn language and task representations for cross-lingual transfer, which allow adaptation to languages not seen during pretraining.

## 5 Conclusion

We evaluate the impact of intermediate-task training on zero-shot cross-lingual transfer. We investigate 9 intermediate tasks and how intermediate-task training impacts the zero-shot cross-lingual transfer to the 9 target tasks in the XTREME benchmark.

Overall, intermediate-task training significantly improves the performance on BUCC and Tatoeba, the two sentence retrieval target tasks in the XTREME benchmark, across almost every intermediate-task configuration. Our best models obtain 5.9 and 23.9 point gains on BUCC and Tatoeba, respectively, compared to the best available XLM-R baseline scores (Hu et al., 2020). We also observed gains in question-answering tasks, particularly using SQuAD v1.1 and v2.0 as intermediate tasks, with absolute gains of 2.1 F1 for XQuAD, 0.8 F1 for MLQA, and 10.4 for F1 TyDiQA, again over the best available baseline scores. We improve over XLM-R by 5.4 points on average on the XTREME benchmark. Additionally, we found multi-task training on all 9 intermedi-

ate tasks to slightly outperform individual intermediate training. On the other hand, we found that neither incorporating multilingual MLM into the intermediate-task training phase nor translating intermediate-task data consistently led to improved transfer performance.

While we have explored the extent to which English intermediate-task training can improve cross-lingual transfer, a clear next avenue of investigation for future work is how the choice of intermediate- and target-task languages influences transfer across different tasks.

## Acknowledgments

This project has benefited from support to SB by Eric and Wendy Schmidt (made by recommendation of the Schmidt Futures program), by Samsung Research (under the project *Improving Deep Learning using Latent Structure*), by Intuit, Inc., by NVIDIA Corporation (with the donation of a Titan V GPU), by Google (with the donation of Google Cloud credits). IC has received funding from the European Union’s Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No 838188. This project has benefited from direct support by the NYU IT High Performance Computing Center. This material is based upon work supported by the National Science Foundation under Grant No. 1922658. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

## References

- Željko Agić. 2017. [Cross-lingual parser selection for low-resource languages](#). In *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies (UDW 2017)*, pages 1–10, Gothenburg, Sweden. Association for Computational Linguistics.
- Željko Agić, Anders Johannsen, Barbara Plank, Héctor Martínez Alonso, Natalie Schluter, and Anders Søgaard. 2016. [Multilingual projection for parsing truly low-resource languages](#). *Transactions of the Association for Computational Linguistics*, 4:301–312.
- Roe Aharoni, Melvin Johnson, and Orhan Firat. 2019. [Massively multilingual neural machine translation](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3874–3884, Minneapolis, Minnesota. Association for Computational Linguistics.
- Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2020a. Translation artifacts in cross-lingual transfer learning. *arXiv preprint arXiv:2004.04721*.
- Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2020b. [On the cross-lingual transferability of monolingual representations](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4623–4637, Online. Association for Computational Linguistics.
- Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.
- Samuel R. Bowman, Gabor Angeli, Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.
- Samuel R Bowman, Jennimaria Palomaki, Livio Baldini Soares, and Emily Pitler. 2020. Collecting entailment data for pretraining: New protocols and negative results. *arXiv preprint arXiv:2004.11997*.
- Jonathan H. Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. 2020. Tydi qa: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*.
- Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. [Unsupervised cross-lingual representation learning at scale](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.
- Alexis Conneau and Guillaume Lample. 2019. [Cross-lingual language model pretraining](#). In H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32 (NeurIPS)*, pages 7059–7069. Curran Associates, Inc.
- Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. [XNLI: Evaluating cross-lingual sentence representations](#). In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.

- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019a. **BERT: Pre-training of deep bidirectional transformers for language understanding**. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019b. **BERT: Pre-training of deep bidirectional transformers for language understanding**. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- Yuwei Fang, Shuohang Wang, Zhe Gan, Siqi Sun, and Jingjing Liu. 2020. **FILTER: An Enhanced Fusion Method for Cross-lingual Language Understanding**. *arXiv e-prints*, page arXiv:2009.05166.
- Julia Hockenmaier and Mark Steedman. 2007. **CCGbank: A corpus of CCG derivations and dependency structures extracted from the Penn treebank**. *Computational Linguistics*, 33(3):355–396.
- Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. **Xtreme: A massively multilingual multi-task benchmark for evaluating cross-lingual generalization**. *Proceedings of the 37th International Conference on Machine Learning*.
- Haoyang Huang, Yaobo Liang, Nan Duan, Ming Gong, Linjun Shou, Dixin Jiang, and Ming Zhou. 2019a. **Unicoder: A universal language encoder by pre-training with multiple cross-lingual tasks**. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2485–2494, Hong Kong, China. Association for Computational Linguistics.
- Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019b. **Cosmos QA: Machine reading comprehension with contextual commonsense reasoning**. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2391–2401, Hong Kong, China. Association for Computational Linguistics.
- Rebecca Hwa, Philip Resnik, Amy Weinberg, Clara Cabezas, and Okan Kolak. 2005. **Bootstrapping parsers via syntactic projection across parallel texts**. *Nat. Lang. Eng.*, 11(3):311–325.
- Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. **Google’s multilingual neural machine translation system: Enabling zero-shot translation**. *Transactions of the Association for Computational Linguistics*, 5:339–351.
- Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, André F. T. Martins, and Alexandra Birch. 2018. **Marian: Fast neural machine translation in C++**. In *Proceedings of ACL 2018, System Demonstrations*, pages 116–121, Melbourne, Australia. Association for Computational Linguistics.
- James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. **Overcoming catastrophic forgetting in neural networks**. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526.
- Ronan Le Bras, Swabha Swayamdipta, Chandra Bhagavatula, Rowan Zellers, Matthew E Peters, Ashish Sabharwal, and Yejin Choi. 2020. **Adversarial filters of dataset biases**. *Proceedings of the 37th International Conference on Machine Learning*.
- Patrick Lewis, Barlas Ouz, Ruty Rinott, Sebastian Riedel, and Holger Schwenk. 2020. **MLQA: Evaluating cross-lingual extractive question answering**. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7315–7330, Online. Association for Computational Linguistics.
- Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019a. **Multi-task deep neural networks for natural language understanding**. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4496, Florence, Italy. Association for Computational Linguistics.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. **Roberta: A robustly optimized bert pretraining approach**. *arXiv preprint arXiv:1907.11692*.
- Stephen Mayhew, Chen-Tse Tsai, and Dan Roth. 2017. **Cheap translation for cross-lingual named entity recognition**. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2536–2545, Copenhagen, Denmark. Association for Computational Linguistics.
- Ryan McDonald, Slav Petrov, and Keith Hall. 2011. **Multi-source transfer of delexicalized dependency parsers**. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 62–72, Edinburgh, Scotland, UK. Association for Computational Linguistics.

- Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.
- Joakim Nivre, Mitchell Abrams, Željko Agić, Lars Ahrenberg, Lene Antonsen, Maria Jesus Aranzabe, Gashaw Arutie, Masayuki Asahara, Luma Ateyah, Mohammed Attia, and et al. 2018. Universal Dependencies 2.2. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.
- Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. 2017. Cross-lingual name tagging and linking for 282 languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1946–1958, Vancouver, Canada. Association for Computational Linguistics.
- Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.
- Jonas Pfeiffer, Ivan Vulic, Iryna Gurevych, and Sebastian Ruder. 2020. MAD-X: An adapter-based framework for multi-task cross-lingual transfer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Jason Phang, Thibault Févry, and Samuel R. Bowman. 2018. Sentence Encoders on STILTs: Supplementary Training on Intermediate Labeled-data Tasks. Unpublished manuscript available on arXiv.
- Jason Phang, Phil Yeres, Jesse Swanson, Haokun Liu, Alex Wang, Ian F. Tenney, Yada Pruksachatkun, Phu Mon Htut, , Katherin Yu, Jan Hula, Patrick Xia, Raghu Pappagari, Shuning Jin, R. Thomas McCoy, Roma Patel, Yinghui Huang, Edouard Grave, Najoung Kim, Thibault Févry, Berlin Chen, Nikita Nangia, Anhad Mohananey, Katharina Kann, Shikha Bordia, Nicolas Patry, David Benton, Ellie Pavlick, and Samuel R. Bowman. 2020. jiant 2.0: A software toolkit for research on general-purpose text understanding models. <http://jiant.info/>.
- Yada Pruksachatkun, Jason Phang, Haokun Liu, Phu Mon Htut, Xiaoyi Zhang, Richard Yuanzhe Pang, Clara Vania, Katharina Kann, and Samuel R. Bowman. 2020. Intermediate-task transfer learning with pretrained language models: When and why does it work? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5231–5247, Online. Association for Computational Linguistics.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.
- Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.
- Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.
- Robert Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-First AAAI Conference on Artificial Intelligence*.
- Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.
- Jörg Tiedemann. 2012. Parallel Data, Tools and Interfaces in OPUS. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. European Language Resources Association (ELRA).
- Jörg Tiedemann and Santhosh Thottingal. 2020. OPUS-MT — Building open translation services for the World. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation (EAMT)*, Lisbon, Portugal.
- Tu Vu, Tong Wang, Tsendsuren Munkhdalai, Alessandro Sordoni, Adam Trischler, Andrew Mattarella-Micke, Subhransu Maji, and Mohit Iyyer. 2020. Exploring and predicting transferability across NLP tasks.
- Alex Wang, Jan Hula, Patrick Xia, Raghavendra Pappagari, R. Thomas McCoy, Roma Patel, Najoung Kim, Ian Tenney, Yinghui Huang, Katherin Yu,

Shuning Jin, Berlin Chen, Benjamin Van Durme, Edouard Grave, Ellie Pavlick, and Samuel R. Bowman. 2019a. [Can you tell me how to get past sesame street? sentence-level pretraining beyond language modeling](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4465–4476, Florence, Italy. Association for Computational Linguistics.

Alex Wang, Jan Hula, Patrick Xia, Raghavendra Papagari, R. Thomas McCoy, Roma Patel, Najoung Kim, Ian Tenney, Yinghui Huang, Katherin Yu, Shuning Jin, Berlin Chen, Benjamin Van Durme, Edouard Grave, Ellie Pavlick, and Samuel R. Bowman. 2019b. [Can you tell me how to get past sesame street? sentence-level pretraining beyond language modeling](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4465–4476, Florence, Italy. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. [A broad-coverage challenge corpus for sentence understanding through inference](#). In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierrette Cistac, Tim Rault, R’emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. [Huggingface’s transformers: State-of-the-art natural language processing](#). Unpublished manuscript available on arXiv.

Yinfei Yang, Yuan Zhang, Chris Tar, and Jason Baldridge. 2019. [PAWS-x: A cross-lingual adversarial dataset for paraphrase identification](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3687–3692, Hong Kong, China. Association for Computational Linguistics.

Dani Yogatama, Cyprien de Masson d’Autume, Jerome Connor, Tomas Kociský, Mike Chrzanowski, Lingpeng Kong, Angeliki Lazaridou, Wang Ling, Lei Yu, Chris Dyer, et al. 2019. Learning and evaluating general linguistic intelligence. *arXiv preprint arXiv:1901.11373*.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. [SWAG: A large-scale adversarial dataset for grounded commonsense inference](#). In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 93–104, Brussels, Belgium. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. [HellaSwag: Can](#)

a machine really finish your sentence?

In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Pierre Zweigenbaum, Serge Sharoff, and Reinhard Rapp. 2017. [Overview of the second BUCC shared task: Spotting parallel sentences in comparable corpora](#). In *Proceedings of the 10th Workshop on Building and Using Comparable Corpora*, pages 60–67, Vancouver, Canada. Association for Computational Linguistics.

Pierre Zweigenbaum, Serge Sharoff, and Reinhard Rapp. 2018. [Overview of the third BUCC shared task:spotting parallel sentences in comparable corpora](#). In *Proceedings of 11th Workshop on Building and Using Comparable Corpora*, pages 39–42.

## A Implementation Details

### A.1 Intermediate Tasks

For intermediate-task training, we use a learning rate of 1e-5 without MLM, and 5e-6 with MLM. Hyperparameters in the Table 4 were chosen based on intermediate task validation performance in a preliminary search. We use a warmup of 10% of the total number of steps, and perform early stopping based on the first 500 development set examples of each task with a patience of 30. For CCG, where tags are assigned for each word, we use the representation of first sub-word token of each word for prediction.

Task	Batch size	# Epochs
ANLI <sup>+</sup>	24	2
MNLI	24	2
CCG	24	15
CommonsenseQA	4	10
Cosmos QA	4	15
HellaSwag	24	7
QQP	24	3
SQuAD	8	3
MLM	8	-
Multi-task	Mixed	3

Table 4: Intermediate-task training configuration.

### A.2 XTREME Benchmark Target Tasks

We follow the sample implementation for the XTREME benchmark unless otherwise stated. We use a learning rate of 3e-6, and use the same optimization procedure as for intermediate tasks. Hyperparameters in the Table 5 follow the sample implementation. For POS and NER, we use the same strategy as for CCG for matching tags to tokens. For BUCC and Tatoeba, we extract the representations for each token from the 13th self-attention layer, and use the mean-pooled representation as the embedding for that example, as in the sample implementation. Similarly, we follow the sample implementation and set an optimal threshold for each language sub-task for BUCC as a similarity score cut-off for extracting parallel sentences based on the development set and applied to the test set.

We randomly initialize the corresponding output heads for each task, regardless of the similarity between intermediate and target tasks (e.g. even if both the intermediate and target tasks train on SQuAD, we randomly initialize the output head in between phases).

Task	Batch size	# Epochs
XNLI (MNLI)	4	2
PAWS-X	32	5
XQuAD (SQuAD)	16	2
MLQA (SQuAD)	16	2
TyDiQA	16	2
POS	32	10
NER	32	10
BUCC	-	-
Tatoeba	-	-

Table 5: Target-task training configuration.

## B Per-Language Results

	ar	bg	de	el	en	es	fr	hi	ru	sw	th	tr	ur	vi	zh	Avg	
XLM-R	<b>79.8</b>	82.7	<b>83.8</b>	81.3	89.3	84.4	83.7	77.3	79.2	72.4	77.1	78.9	72.6	80.0	79.6	80.1	
Without MLM	ANLI <sup>+</sup>	77.5	82.5	82.3	80.8	87.6	83.5	83.6	76.5	79.1	70.4	77.3	78.0	73.5	79.2	79.3	79.4
	MNLI	78.4	<b>82.8</b>	83.0	81.3	88.2	84.0	<b>83.6</b>	77.2	<b>79.5</b>	69.4	77.6	77.9	73.2	79.8	79.1	79.7
	QQP	77.1	81.0	81.6	<b>81.6</b>	86.1	83.6	82.0	75.4	78.5	69.6	76.9	77.1	72.7	79.2	78.6	78.7
	SQuAD v2.0	77.9	81.3	81.7	79.9	85.6	83.5	81.8	75.5	78.5	70.6	77.2	77.2	<b>73.7</b>	78.9	<b>79.6</b>	78.9
	SQuAD v1.1	77.1	82.1	81.8	79.9	87.1	82.8	82.7	75.5	78.6	71.3	76.3	77.3	71.2	79.2	78.6	78.8
	HellaSwag	<b>78.6</b>	82.6	<b>83.5</b>	80.6	<b>88.5</b>	83.7	83.1	<b>77.4</b>	78.2	<b>72.0</b>	<b>77.4</b>	<b>78.7</b>	73.5	<b>80.0</b>	79.4	<b>79.8</b>
	CCG	77.3	81.9	81.7	79.8	88.1	82.9	83.2	75.4	78.8	69.9	<b>76.5</b>	76.9	71.4	79.7	78.6	78.8
	Cosmos QA	77.1	81.1	81.7	80.1	87.4	83.2	81.7	74.3	77.7	<b>72.0</b>	75.2	76.7	71.1	78.3	78.4	78.4
	CSQA	77.3	80.8	81.9	80.0	87.5	83.5	82.5	76.3	78.4	70.6	76.3	<b>77.5</b>	72.5	79.6	78.5	78.9
	Multi-task	76.9	82.2	82.9	81.0	88.5	84.4	82.5	75.8	79.1	71.1	77.1	79.1	72.0	79.6	79.2	79.4
With MLM	ANLI <sup>+</sup>	78.5	82.8	<b>83.8</b>	81.5	89.2	84.1	82.5	76.5	79.2	72.7	77.4	78.6	72.7	80.7	80.1	80.0
	MNLI	78.0	82.9	83.1	81.1	88.8	84.3	83.4	76.7	<b>80.3</b>	72.2	<b>78.4</b>	79.3	73.4	80.5	80.2	80.2
	QQP	78.0	81.7	83.3	80.8	88.6	<b>84.5</b>	82.9	75.9	78.3	72.2	77.7	78.6	72.7	79.9	78.9	79.6
	SQuAD v2.0	77.5	82.8	83.3	80.4	88.8	83.6	82.7	76.0	79.6	71.6	77.0	78.7	72.9	79.9	78.9	79.6
	SQuAD v1.1	77.9	81.7	82.2	79.7	87.0	82.8	82.1	74.4	78.4	71.2	76.6	78.1	71.3	79.0	78.6	78.7
	HellaSwag	<b>79.3</b>	<b>83.5</b>	83.7	<b>81.8</b>	<b>89.6</b>	<b>84.5</b>	<b>84.1</b>	<b>78.2</b>	79.9	72.9	78.1	<b>80.1</b>	<b>74.5</b>	<b>81.3</b>	<b>80.7</b>	<b>80.8</b>
	CCG	77.9	82.5	82.4	80.8	87.1	83.8	82.6	76.6	78.9	72.0	76.7	78.2	72.2	80.2	78.4	79.4
	Cosmos QA	78.1	82.7	82.7	80.4	87.6	83.9	82.9	76.2	79.5	<b>73.7</b>	77.8	79.0	72.7	80.4	79.6	79.8
	CSQA	79.0	83.4	83.7	81.2	89.0	83.8	83.3	76.9	79.9	72.3	78.0	79.1	73.3	80.4	80.6	80.2

Table 6: Full XNLI Results

	de	en	es	fr	ja	ko	zh	Avg	
XLM-R	88.1	93.4	89.2	89.3	81.8	81.8	82.0	86.5	
Without MLM	ANLI <sup>+</sup>	88.0	94.1	89.6	90.7	82.0	82.2	81.9	87.0
	MNLI	89.0	95.0	90.7	90.9	82.9	83.8	84.2	88.1
	QQP	83.9	93.0	87.7	88.7	79.2	78.6	79.7	84.4
	SQuADv2.0	88.9	<b>95.2</b>	<b>91.7</b>	<b>91.3</b>	84.7	84.5	<b>85.4</b>	<b>88.8</b>
	SQuADv1.1	<b>89.4</b>	94.2	91.1	91.1	83.8	83.5	<b>83.9</b>	88.1
	HellaSwag	88.4	95.0	90.2	91.1	<b>84.8</b>	<b>84.6</b>	84.5	88.4
	CCG	83.5	92.3	86.5	88.1	<b>78.0</b>	<b>77.0</b>	78.6	83.5
	Cosmos QA	88.4	93.8	90.4	90.3	84.3	84.3	85.0	88.1
	CSQA	85.9	93.7	88.6	89.8	81.7	80.4	81.5	86.0
	Multi-task	89.0	95.0	90.2	91.1	83.8	83.5	85.5	88.3
With MLM	ANLI <sup>+</sup>	88.1	94.5	90.1	90.4	84.0	84.2	84.2	87.9
	MNLI	<b>90.1</b>	<b>95.5</b>	91.3	91.3	84.4	84.1	84.5	88.7
	QQP	88.6	94.3	89.8	90.6	81.7	82.8	82.3	87.1
	SQuADv2.0	88.9	95.0	<b>91.7</b>	<b>92.0</b>	<b>85.2</b>	83.9	84.7	88.8
	SQuADv1.1	89.0	93.8	90.3	88.9	82.7	82.2	82.2	87.0
	HellaSwag	<b>90.3</b>	95.0	91.0	90.5	84.9	<b>85.9</b>	<b>84.8</b>	<b>88.9</b>
	CCG	87.5	93.3	88.3	88.4	81.5	81.2	81.3	85.9
	Cosmos QA	88.1	94.0	89.4	90.0	82.5	82.4	82.3	87.0
	CSQA	88.7	94.1	89.1	89.8	82.5	82.9	82.2	87.0

Table 7: Full PAWS-X Results

	af	ar	bg	de	el	en	es	et	eu	fa	fi	fr	he	hi	hu	id	it	
XLM-R	87.7	56.3	87.9	88.6	85.6	95.9	89.8	87.6	72.8	70.0	84.9	<b>65.5</b>	68.1	73.2	81.3	<b>81.7</b>	88.8	
Without MLM	ANLI <sup>+</sup>	87.9	57.6	88.3	88.8	85.6	95.7	89.4	87.3	73.4	72.0	84.9	65.4	70.9	70.1	<b>82.9</b>	81.0	88.3
	MNLI	87.9	56.6	87.8	88.5	84.6	96.2	88.9	86.9	70.4	69.5	84.1	51.8	70.1	72.4	81.2	81.1	88.6
	QQP	83.9	52.6	86.0	85.3	81.7	93.7	87.7	82.1	70.1	66.7	79.3	62.5	61.1	62.5	78.3	79.2	86.8
	SQuADv2.0	87.5	58.0	88.0	87.9	83.6	96.2	88.7	86.6	69.9	69.1	83.9	51.8	<b>71.3</b>	69.7	82.6	81.0	89.0
	SQuADv1.1	87.7	<b>58.1</b>	<b>88.6</b>	88.4	85.8	95.7	<b>89.4</b>	87.2	73.4	70.1	84.3	65.1	70.9	72.2	81.8	<b>81.3</b>	88.5
	HellaSwag	88.3	57.3	88.5	88.7	85.6	<b>96.5</b>	89.2	87.6	72.6	69.5	84.7	52.5	69.6	74.8	81.6	<b>81.1</b>	<b>89.6</b>
	CCG	88.2	56.2	86.5	<b>89.4</b>	85.9	95.8	87.8	87.9	73.7	69.1	<b>85.6</b>	53.5	68.8	75.1	81.8	80.8	86.8
	Cosmos QA	<b>88.4</b>	56.4	86.2	88.0	84.4	95.9	88.9	87.1	73.5	71.2	84.5	65.3	67.5	<b>75.6</b>	81.1	81.0	88.8
	CSQA	87.1	55.7	87.6	87.8	85.8	95.4	88.6	87.3	<b>76.4</b>	69.3	84.7	64.6	65.3	67.6	81.2	80.9	86.6
	Multi-task	87.7	58.5	89.7	88.8	85.2	96.3	89.4	87.1	67.7	71.6	84.7	52.7	71.0	68.2	81.5	80.7	89.8
With MLM	ANLI <sup>+</sup>	87.9	<b>58.4</b>	88.3	88.9	<b>86.3</b>	95.8	<b>90.3</b>	87.8	<b>76.4</b>	<b>72.5</b>	<b>85.1</b>	53.3	69.0	72.5	<b>82.4</b>	80.7	88.6
	MNLI	<b>89.1</b>	57.2	87.6	88.6	85.1	96.2	88.8	88.0	73.4	69.5	<b>85.1</b>	52.7	68.0	76.9	80.6	80.4	88.7
	QQP	87.7	56.3	87.6	88.6	84.2	95.9	89.6	<b>88.1</b>	76.3	71.2	84.5	59.7	67.5	<b>78.0</b>	81.8	81.2	88.8
	SQuADv2.0	88.5	57.8	87.8	88.5	85.8	96.2	89.0	<b>86.1</b>	74.7	71.0	84.6	49.1	68.2	73.2	81.4	80.8	85.8
	SQuADv1.1	88.0	55.1	<b>88.6</b>	88.9	85.3	95.7	89.7	85.7	73.5	70.2	83.5	64.5	66.7	74.4	79.7	<b>81.5</b>	86.8
	HellaSwag	88.3	58.0	87.8	88.3	85.7	<b>96.4</b>	87.2	86.8	74.0	70.2	84.3	51.5	<b>70.9</b>	74.8	79.9	81.0	88.4
	CCG	88.1	54.5	86.7	89.2	<b>86.3</b>	95.9	87.5	87.6	77.2	71.4	84.0	64.4	66.3	76.7	81.1	81.4	<b>89.0</b>
	Cosmos QA	87.5	57.8	87.7	88.6	85.5	95.8	89.5	<b>88.1</b>	71.7	70.1	84.9	64.4	68.9	76.6	81.0	80.0	88.3
	CSQA	87.6	55.9	87.4	88.7	85.1	95.6	88.5	87.2	<b>76.4</b>	70.4	84.2	<b>65.1</b>	68.2	68.3	81.6	81.2	88.4
	Multi-task	87.7	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	
Without MLM	ja	kk	ko	mr	nl	pt	ru	ta	te	th	tl	tr	ur	vi	yo	zh	Avg	
	XLM-R	31.9	-	50.4	80.0	90.1	<b>90.2</b>	89.5	67.1	<b>90.0</b>	-	-	76.0	65.6	56.4	-	40.9	75.7
	ANLI <sup>+</sup>	19.4	-	50.7	79.6	90.1	89.7	<b>90.0</b>	69.2	86.6	-	-	75.0	66.2	55.3	-	27.2	74.8
	MNLI	38.1	-	50.7	79.1	90.4	89.7	89.4	69.4	86.7	-	-	74.8	67.6	54.4	-	<b>48.6</b>	75.4
	QQP	6.2	-	45.9	73.5	88.4	88.2	86.6	65.1	81.7	-	-	71.5	59.1	54.5	-	12.0	70.1
	SQuADv2.0	<b>39.4</b>	-	50.8	80.5	90.3	<b>90.1</b>	89.1	68.5	86.1	-	-	74.1	60.6	54.1	-	45.3	75.0
	SQuADv1.1	30.9	-	49.7	78.7	<b>90.5</b>	89.7	89.3	66.8	84.9	-	-	74.4	65.4	56.2	-	37.7	75.3
	HellaSwag	31.1	-	50.5	83.7	90.1	89.8	89.5	<b>69.7</b>	86.2	-	-	74.2	67.4	54.5	-	35.1	75.2
	CCG	17.8	-	50.3	81.0	90.1	88.0	88.9	66.8	88.4	-	-	75.9	70.7	55.5	-	23.1	74.1
	Cosmos QA	16.4	-	50.3	77.7	89.9	89.7	89.4	67.9	88.1	-	-	<b>76.5</b>	<b>69.2</b>	<b>56.3</b>	-	23.2	74.4
With MLM	CSQA	32.4	-	49.3	<b>82.8</b>	89.4	88.5	88.5	66.9	86.3	-	-	74.5	<b>63.5</b>	<b>56.0</b>	-	29.6	74.5
	Multi-task	36.4	-	50.7	79.6	90.0	89.8	88.9	68.4	86.2	-	-	74.4	62.2	55.5	-	44.3	75.1
	ANLI <sup>+</sup>	<b>39.0</b>	-	51.2	80.7	90.2	<b>90.0</b>	<b>89.8</b>	68.7	87.6	-	-	<b>76.4</b>	66.2	56.7	-	<b>45.7</b>	<b>76.1</b>
	MNLI	30.1	-	51.0	80.1	90.0	<b>88.8</b>	<b>89.1</b>	<b>68.8</b>	<b>85.5</b>	-	-	<b>75.1</b>	69.6	55.4	-	38.4	75.1
	QQP	27.6	-	50.8	81.0	90.1	89.5	89.4	67.2	<b>88.0</b>	-	-	<b>76.2</b>	<b>70.3</b>	56.5	-	34.0	75.4
	SQuADv2.0	35.3	-	51.0	80.2	89.9	88.1	89.3	67.1	84.3	-	-	75.5	68.8	56.9	-	39.0	75.0
	SQuADv1.1	16.3	-	49.7	79.4	90.2	<b>90.0</b>	89.2	68.0	83.3	-	-	75.8	64.6	<b>57.3</b>	-	19.0	73.8
	HellaSwag	35.4	-	50.9	78.4	90.0	87.9	89.3	68.7	86.4	-	-	75.4	69.3	54.8	-	43.6	75.3
	CCG	25.7	-	50.7	<b>86.1</b>	89.8	88.8	88.4	68.0	86.6	-	-	76.2	68.2	55.5	-	23.9	75.0
	Cosmos QA	16.5	-	51.0	80.9	89.7	88.9	89.0	67.4	87.9	-	-	76.3	70.1	56.0	-	19.6	74.5
	CSQA	30.8	-	<b>51.8</b>	80.5	<b>90.5</b>	89.6	89.0	66.8	86.5	-	-	74.8	61.9	56.3	-	31.3	74.8

Table 8: Full POS Results. kk, th, tl and yo do not have development set data.

	af	ar	bg	bn	de	el	en	es	et	eu	fa	fi	fr	he	hi	hu	id	it	ja	jv	ka	
XLM-R	77.7	47.1	81.9	74.9	78.6	76.3	81.6	74.7	77.2	61.2	58.2	78.3	78.3	50.2	68.7	80.6	53.7	80.8	15.6	56.2	61.4	
Without MLM	ANLI <sup>+</sup>	75.4	52.7	78.1	72.7	76.4	76.3	80.9	71.6	72.8	52.2	60.7	75.8	77.4	49.1	69.6	79.6	52.7	78.9	13.1	54.3	62.1
	MNLI	76.9	48.3	80.5	72.8	77.7	77.9	84.2	76.9	78.5	62.1	58.3	78.7	81.1	55.1	69.0	81.1	55.7	80.8	16.4	54.2	68.1
	QQP	73.8	40.9	75.5	66.0	71.3	71.6	75.8	65.5	69.3	55.5	49.9	73.1	72.8	42.6	59.8	74.3	49.2	75.9	5.7	54.4	51.1
	SQuADv2.0	76.0	48.0	81.1	71.8	78.4	78.2	84.3	74.7	78.4	53.9	56.9	78.9	<b>82.5</b>	56.0	68.9	79.8	56.4	80.8	18.1	61.8	67.3
	SQuADv1.1	<b>79.1</b>	52.6	80.1	75.5	77.8	78.1	80.8	75.3	76.7	54.3	61.9	78.7	78.4	52.8	65.6	80.3	54.6	80.8	18.7	52.1	62.4
	HellaSwag	77.0	54.9	82.7	<b>76.6</b>	79.1	<b>78.9</b>	84.3	<b>77.8</b>	78.0	58.8	65.0	77.5	80.3	<b>57.0</b>	<b>71.2</b>	81.8	54.3	<b>81.4</b>	19.6	56.9	70.6
	CCG	77.4	51.5	78.7	72.5	78.4	76.2	80.8	73.0	78.0	56.9	62.1	78.2	77.3	48.6	67.3	79.7	54.9	79.9	15.9	60.3	58.9
	Cosmos QA	76.6	49.3	79.2	76.0	77.8	76.1	81.2	73.2	76.6	59.8	55.8	77.8	77.0	46.8	67.8	79.4	53.2	80.0	14.1	55.5	57.8
	CSQA	77.6	46.1	78.9	75.4	78.4	76.2	81.3	77.3	75.2	59.8	61.9	78.0	78.2	48.9	67.6	79.6	55.6	80.1	11.6	53.8	57.7
	Multi-task	78.5	49.2	82.0	73.3	78.9	80.1	84.5	76.6	78.5	59.4	49.4	79.1	81.2	56.4	70.6	81.0	57.0	80.7	20.7	64.7	68.6
With MLM	ANLI <sup>+</sup>	76.4	51.5	80.7	73.3	79.2	77.8	84.3	75.4	78.0	57.7	49.7	77.6	80.1	54.8	68.9	80.8	54.8	80.5	14.4	54.9	64.5
	MNLI	78.0	52.3	81.7	73.0	78.1	84.4	77.2	79.4	59.6	60.6	79.2	81.4	55.1	68.6	81.0	51.3	81.0	14.0	62.0	64.3	
	QQP	77.1	46.7	79.0	72.9	79.4	76.3	81.9	74.2	78.7	61.8	<b>66.0</b>	78.3	78.0	50.4	69.1	81.6	53.2	80.1	15.1	<b>62.6</b>	60.7
	SQuADv2.0	78.0	46.5	<b>82.8</b>	71.7	79.0	77.3	84.2	74.8	79.0	61.6	63.3	79.5	80.0	<b>57.6</b>	67.5	<b>81.9</b>	<b>62.0</b>	80.7	<b>20.0</b>	62.3	68.2
	SQuADv1.1	<b>77.7</b>	<b>58.0</b>	81.4	75.2	78.0	77.4	82.1	69.6	76.1	54.1	58.4	77.5	78.7	54.8	67.5	78.8	49.9	79.5	14.5	55.9	<b>68.3</b>
	HellaSwag	78.7	47.0	81.8	73.8	<b>79.7</b>	78.2	<b>84.8</b>	73.6	79.2	55.8	55.6	78.2	79.4	55.0	69.8	81.3	54.1	81.3	18.5	58.1	67.5
	CCG	74.5	46.4	76.7	74.5	76.9	75.7	80.5	72.6	77.7	58.9	59.6	77.7	77.0	48.1	66.3	80.1	53.4	78.7	13.8	57.1	58.2
	Cosmos QA	78.2	39.1	80.0	73.8	79.0	77.2	81.4	70.3	78.8	65.4	48.9	78.7	77.7	48.3	68.0	80.8	55.1	81.2	13.2	58.9	59.0
	CSQA	77.4	48.8	78.9	73.9	78.8	76.3	81.9	75.2	<b>79.5</b>	<b>66.7</b>	58.6	78.5	47.7	68.2	81.0	55.3	81.3	12.2	60.4	58.9	
Without MLM	kk	ko	ml	mr	ms	my	nl	pt	ru	sw	ta	te	th	tl	tr	ur	vi	yo	zh	Avg		
	XLM-R	48.7	54.5	58.8	61.8	54.1	53.7	83.2	80.7	69.3	69.8	58.2	50.8	2.2	73.2	81.1	67.0	74.9	33.2	23.6	62.8	-
	ANLI <sup>+</sup>	50.2	52.6	61.2	63.0	66.8	46.5	81.8	78.7	67.0	66.9	55.0	52.1	2.5	71.2	78.0	67.3	73.9	43.3	18.9	62.0	-
	MNLI	51.7	58.8	64.8	61.3	69.8	54.9	83.0	80.8	70.2	70.3	59.3	<b>55.4</b>	1.0	74.8	80.5	56.9	78.1	38.9	25.2	64.2	-
	QQP	50.4	40.1	51.2	51.4	61.4	32.5	78.2	73.0	50.8	65.1	47.3	41.4	1.6	67.4	72.3	57.2	67.9	43.9	8.6	55.9	-
	SQuADv2.0	49.9	58.1	61.6	62.5	72.1	50.0	83.1	82.3	70.8	65.4	62.6	53.6	0.6	74.8	80.0	63.2	<b>78.9</b>	41.2	22.5	64.1	-
	SQuADv1.1	51.8	57.1	61.7	59.8	50.4	52.2	83.3	80.8	69.8	69.2	58.3	49.5	0.8	71.6	79.1	58.6	76.3	<b>47.5</b>	26.2	63.0	-
	HellaSwag	50.5	58.4	56.6	<b>66.6</b>	72.8	<b>59.4</b>	83.2	<b>82.5</b>	70.8	69.9	<b>63.7</b>	53.0	1.1	75.1	78.0	70.0	75.0	42.1	<b>29.7</b>	<b>65.5</b>	-
	CCG	52.4	52.7	57.7	59.6	52.3	50.0	82.5	79.0	67.1	67.0	55.3	49.1	<b>2.6</b>	70.0	81.0	65.3	74.2	37.6	23.3	62.1	-
	Cosmos QA	48.4	52.4	60.3	62.1	56.9	50.2	82.8	79.5	67.4	67.8	57.2	51.4	1.3	74.6	80.7	60.8	74.9	34.8	19.5	61.8	-
	CSQA	49.7	52.0	59.1	62.9	62.4	46.1	82.5	80.3	65.4	69.0	57.1	51.2	1.8	73.1	80.2	<b>73.3</b>	73.5	35.3	19.3	62.3	-
	Multi-task	53.2	57.8	60.8	61.0	69.3	54.2	83.8	80.8	69.4	70.6	58.9	53.7	2.2	75.2	77.7	57.7	75.6	46.1	30.4	64.7	-
With MLM	ANLI <sup>+</sup>	52.9	56.8	60.0	61.1	<b>75.4</b>	49.5	<b>83.4</b>	80.9	68.3	71.0	57.2	49.8	0.9	74.5	79.0	59.8	76.3	31.7	22.5	63.2	-
	MNLI	<b>54.7</b>	57.5	63.5	63.3	66.3	49.6	<b>83.4</b>	81.1	70.3	72.2	57.0	53.5	1.1	74.1	80.9	61.1	75.1	43.4	22.8	64.3	-
	QQP	49.9	54.5	63.3	64.6	54.7	49.0	82.9	78.9	68.7	70.9	58.0	50.7	1.1	74.0	<b>82.3</b>	70.2	77.1	40.3	24.9	63.5	-
	SQuADv2.0	52.1	<b>60.8</b>	<b>65.1</b>	63.2	54.7	54.8	<b>83.4</b>	80.9	<b>71.6</b>	<b>72.6</b>	63.0	54.1	0.4	<b>75.3</b>	80.4	59.8	77.6	33.6	28.0	64.7	-
	SQuADv1.1	51.6	57.7	62.7	60.2	62.2	52.9	81.8	77.7	71.4	68.5	59.7	49.9	1.5	72.9	78.1	54.2	71.5	34.3	22.4	62.6	-
	HellaSwag	53.6	58.9	62.5	63.2	72.4	54.7	82.8	80.9	71.3	70.6	59.5	52.0	2.4	73.6	80.1	58.4	78.3	36.8	24.9	64.2	-
	CCG	54.6	53.5	60.6	62.8	69.1	41.6	80.7	78.1	65.4	68.1	55.1	51.6	1.3	68.7	79.8	61.9	68.8	37.9	19.8	61.6	-
	Cosmos QA	49.7	52.5	55.7	60.2	52.1	48.1	82.9	78.9	67.1	66.6	55.3	47.7	0.9	74.7	80.8	59.5	74.0	34.9	19.3	61.3	-
	CSQA	52.2	54.4	60.4	61.1	52.9	47.8	<b>83.4</b>	80.7	68.5	69.0	57.9	50.1	1.4	73.6	81.5	63.2	74.0	43.6	19.3	62.9	-

Table 9: Full NER Results

	ar	de	el	en	es	hi	rr	th	tr	vi	zh	Avg	
XLM-R	72.5 / 53.4	77.7 / 61.2	77.6 / 59.2	86.3 / 74.2	80.0 / 61.0	73.7 / 57.5	77.7 / 59.8	72.8 / 62.3	72.6 / 54.8	77.6 / 58.0	68.7 / 58.2	76.1 / 60.0	
Without MLM	ANLI <sup>+</sup>	72.9 / 55.0	77.2 / 60.7	75.8 / 58.3	84.9 / 73.1	78.4 / 59.5	73.1 / 56.9	76.8 / 59.9	73.0 / 63.3	72.1 / 55.0	78.0 / 57.6	68.3 / 59.0	75.5 / 59.8
	MNLI	70.7 / 53.2	77.4 / 60.2	76.8 / 59.1	84.2 / 72.6	80.3 / 62.5	72.2 / 55.9	77.8 / 61.3	72.9 / 63.5	71.9 / 56.3	78.1 / 59.7	68.0 / 60.0	75.5 / 60.4
	QQP	68.4 / 50.4	73.2 / 56.5	73.3 / 55.9	82.3 / 70.6	75.4 / 57.3	68.5 / 52.5	74.2 / 57.5	68.6 / 60.2	68.3 / 51.4	72.9 / 53.4	66.3 / 58.0	72.0 / 56.7
	SQuADv2.0	72.1 / 56.0	79.5 / 62.0	78.6 / 60.6	86.7 / 75.5	81.5 / <b>63.6</b>	72.7 / 56.2	79.2 / 61.8	71.0 / 56.8	75.0 / 59.1	78.6 / 58.9	68.8 / 57.6	76.9 / 60.7
	SQuADv1.1	<b>75.9 / 59.9</b>	<b>80.3 / 63.6</b>	<b>80.3 / 62.1</b>	<b>88.3 / 77.4</b>	<b>81.8 / 63.2</b>	<b>76.1 / 59.2</b>	<b>80.0 / 64.1</b>	<b>75.6 / 65.5</b>	<b>75.8 / 59.2</b>	<b>80.5 / 61.2</b>	<b>78.7 / 63.3</b>	<b>78.7 / 63.3</b>
	HellaSwag	73.9 / 56.9	78.7 / 61.3	77.9 / 58.8	86.1 / 75.6	79.6 / 60.1	74.3 / 57.5	78.5 / 62.8	73.6 / 64.5	73.5 / 56.6	78.8 / 59.1	69.2 / 59.4	76.7 / 61.1
	CCG	71.5 / 54.2	76.3 / 58.5	75.9 / 58.2	84.2 / 72.3	79.0 / 60.1	72.3 / 54.9	76.7 / 60.0	71.2 / 60.9	71.7 / 55.3	76.4 / 56.9	67.9 / 58.2	74

	ar	de	en	es	hi	vi	zh	Avg
XLM-R	62.7 / 42.4	69.1 / 52.0	81.6 / 68.6	72.2 / 53.0	68.0 / 50.7	69.5 / 47.6	67.9 / 46.2	70.1 / 51.5
Without MLM	ANLI <sup>+</sup>	64.1 / 43.9	66.8 / 49.8	82.5 / 69.4	71.9 / 52.6	69.2 / 50.5	70.5 / 49.7	66.9 / 44.8
	MNLI	64.2 / 43.5	68.1 / 51.8	82.7 / 70.0	73.7 / 54.8	70.3 / 52.7	68.9 / 49.5	67.1 / 46.0
	QQP	60.5 / 39.7	62.4 / 45.5	79.0 / 66.0	70.7 / 51.6	62.9 / 45.4	67.0 / 47.6	63.5 / 41.1
	SQuADv2.0	66.1 / 45.3	68.2 / 50.2	83.5 / <b>71.1</b>	73.6 / 55.4	68.5 / 51.5	71.7 / <b>52.4</b>	68.2 / 46.4
	SQuADv1.1	<b>67.4 / 46.4</b>	<b>69.6 / 52.9</b>	<b>84.1 / 70.8</b>	<b>75.3 / 56.8</b>	<b>72.5 / 54.8</b>	70.9 / 51.7	<b>69.4 / 47.0</b>
	HellaSwag	64.2 / 43.1	68.8 / 52.3	83.5 / 70.9	73.0 / 53.6	69.2 / 51.7	69.8 / 48.7	68.5 / 46.2
	CCG	62.7 / 41.6	67.5 / 50.4	82.9 / 70.0	72.9 / 54.6	66.1 / 50.1	68.9 / 48.9	66.4 / 45.6
	Cosmos QA	63.8 / 43.9	68.2 / 50.4	82.2 / 69.0	72.9 / 54.2	69.4 / 51.7	70.8 / 50.1	66.6 / 44.4
	CSQA	64.0 / 43.9	68.8 / 52.0	83.4 / 70.6	75.2 / 55.0	69.1 / 51.5	<b>72.6 / 52.1</b>	69.2 / 46.6
	Multi-task	65.1 / 44.1	70.2 / 54.9	82.9 / 69.4	75.2 / 56.4	70.1 / 52.3	72.0 / 51.7	68.6 / 46.2
With MLM	ANLI <sup>+</sup>	62.7 / 41.8	68.5 / 51.4	82.1 / 69.0	73.6 / 54.2	66.7 / 48.7	69.5 / 49.3	66.2 / 44.2
	MNLI	62.9 / 41.0	<b>69.2 / 53.5</b>	82.6 / 69.4	74.3 / 54.4	68.0 / 50.7	<b>70.5 / 50.5</b>	68.0 / 45.8
	QQP	64.6 / 44.9	68.1 / 51.2	83.2 / 70.4	74.0 / 55.6	70.4 / 53.1	69.1 / 49.3	68.3 / 45.6
	SQuADv2.0	64.7 / 43.9	66.6 / 51.0	82.1 / 69.6	73.1 / 55.2	70.2 / 53.1	69.0 / 51.1	<b>68.6 / 47.2</b>
	SQuADv1.1	64.4 / 43.3	68.0 / 50.0	83.1 / 70.0	75.2 / 56.2	68.5 / 51.9	71.2 / 51.9	66.8 / 44.6
	HellaSwag	64.7 / 44.3	68.4 / 52.3	83.3 / 70.4	73.9 / 55.0	69.5 / 52.1	69.9 / 47.9	67.7 / 44.8
	CCG	60.4 / 41.4	66.5 / 50.8	81.8 / 68.6	72.8 / 54.2	66.2 / 48.7	67.7 / 46.2	64.5 / 44.6
	Cosmos QA	63.4 / 43.1	69.0 / 51.0	81.9 / 68.9	72.3 / 53.6	66.3 / 48.9	69.1 / 47.6	66.0 / 45.2
	CSQA	64.3 / 43.7	69.5 / 51.8	82.6 / 69.4	73.4 / 54.4	68.0 / 50.7	70.9 / 48.7	67.7 / 45.8
	Multi-task							70.9 / 52.1

Table 11: Full MLQA Results

	ar	bn	en	fi	id	ko	ru	sw	te	Avg
XLM-R	64.5 / 46.9	59.5 / 41.6	70.4 / 56.6	64.9 / 49.2	75.1 / 59.8	54.7 / 39.5	65.4 / 43.6	67.2 / 48.7	68.8 / 48.3	65.6 / 48.2
Without MLM	ANLI <sup>+</sup>	67.3 / 47.8	54.9 / 37.2	71.0 / 57.3	64.7 / 47.8	74.9 / 57.5	54.5 / 41.3	62.4 / 33.0	67.2 / 47.3	68.2 / 46.9
	MNLI	67.8 / 49.7	60.6 / 40.7	71.6 / 57.7	66.5 / 48.6	76.6 / 61.9	55.3 / 42.4	63.9 / 39.0	66.9 / 48.5	71.0 / 51.4
	QQP	63.2 / 44.4	43.8 / 26.5	64.4 / 52.7	56.3 / 39.9	71.6 / 57.0	47.5 / 32.6	57.4 / 38.2	54.5 / 36.5	45.5 / 26.2
	SQuADv2.0	76.5 / 59.8	77.7 / 63.7	76.1 / 63.2	78.3 / 64.3	83.1 / 69.9	68.1 / 56.5	73.0 / 51.5	79.1 / 67.1	79.2 / 61.1
	SQuADv1.1	76.1 / 60.0	75.6 / 61.9	77.6 / 66.6	76.0 / 61.3	82.5 / 68.3	63.7 / 51.4	71.1 / 44.7	76.5 / 63.5	79.0 / 61.6
	HellaSwag	69.9 / 49.4	60.6 / 42.5	72.2 / 59.1	63.0 / 44.1	76.7 / 60.4	54.7 / 39.1	61.4 / 33.0	66.3 / 48.3	70.6 / 47.8
	CCG	63.6 / 41.8	54.1 / 37.2	68.5 / 55.9	59.6 / 41.7	73.2 / 57.5	50.8 / 37.7	60.2 / 33.4	66.8 / 49.7	66.2 / 43.8
	Cosmos QA	71.7 / 51.9	65.9 / 48.7	73.3 / 61.6	66.7 / 50.9	78.5 / 63.4	52.6 / 36.6	66.2 / 44.1	68.0 / 51.3	74.5 / 54.7
	CSQA	70.9 / 52.1	67.8 / 49.6	74.6 / 60.9	69.6 / 52.6	77.0 / 60.2	60.8 / 46.4	63.6 / 36.0	70.8 / 53.5	73.3 / 54.7
	Multi-task	73.3 / 52.3	66.7 / 48.7	75.6 / 63.6	74.7 / 59.6	81.7 / 67.3	60.2 / 46.4	71.0 / 43.0	76.0 / 64.3	77.2 / 58.4
With MLM	ANLI <sup>+</sup>	67.1 / 48.9	59.5 / 42.5	72.2 / 58.9	67.2 / 51.4	76.8 / 60.7	54.9 / 42.0	62.4 / 35.3	70.3 / 52.1	70.4 / 53.1
	MNLI	67.3 / 49.7	60.0 / 41.6	71.2 / 59.3	66.8 / 50.4	78.1 / 62.1	56.4 / 42.0	62.2 / 33.9	68.5 / 50.7	70.0 / 48.4
	QQP	67.8 / 49.0	55.7 / 37.2	69.8 / 56.1	64.1 / 47.1	74.2 / 58.6	49.0 / 34.4	60.0 / 34.5	64.5 / 45.7	70.1 / 45.6
	SQuADv2.0	76.9 / 60.5	70.1 / 54.9	76.6 / 64.5	74.4 / 59.6	83.4 / 69.7	61.6 / 48.6	71.3 / 45.2	74.0 / 61.5	76.7 / 59.3
	SQuADv1.1	77.0 / 59.3	68.5 / 51.3	75.4 / 64.3	77.2 / 63.4	83.3 / 71.0	63.7 / 51.8	71.7 / 47.9	73.1 / 56.5	76.4 / 59.0
	HellaSwag	68.8 / 50.4	62.6 / 47.8	70.9 / 56.8	64.0 / 48.6	77.4 / 61.8	54.6 / 40.9	61.2 / 31.7	68.2 / 49.5	71.4 / 50.5
	CCG	68.1 / 49.1	57.5 / 39.8	69.0 / 55.9	65.9 / 48.6	76.5 / 61.9	55.0 / 39.9	61.6 / 31.9	67.5 / 49.3	56.3 / 30.3
	Cosmos QA	66.6 / 46.6	56.8 / 37.2	71.5 / 58.0	64.2 / 45.0	75.0 / 57.0	56.3 / 41.3	63.6 / 39.0	69.0 / 51.1	63.6 / 46.3
	CSQA	68.8 / 50.4	60.2 / 43.4	71.3 / 59.1	67.6 / 50.5	76.9 / 59.8	54.0 / 41.3	63.5 / 38.1	69.5 / 52.9	72.8 / 54.1
	Multi-task									67.2 / 49.9

Table 12: Full TyDiQA Results

		de	fr	ru	zh	Avg
	XLM-R	77.7	62.7	79.2	66.5	71.5
Without MLM	ANLI <sup>+</sup>	94.6	89.8	93.5	88.6	91.6
	MNLI	94.2	90.2	93.5	<b>89.9</b>	92.0
	QQP	94.2	91.0	93.3	88.5	91.8
	SQuADv2.0	94.0	89.8	93.0	<b>89.9</b>	91.7
	SQuADv1.1	94.2	90.5	93.1	87.0	91.2
	HellaSwag	94.6	<b>91.9</b>	<b>93.9</b>	88.9	<b>92.3</b>
	CCG	88.3	82.9	86.6	78.0	83.9
	Cosmos QA	94.1	90.2	93.2	88.6	91.5
	CSQA	<b>95.1</b>	90.6	93.5	89.1	92.1
	Multi-task	94.3	90.4	93.4	87.0	91.3
With MLM	ANLI <sup>+</sup>	93.4	88.0	92.9	86.5	90.2
	MNLI	92.7	89.0	93.2	86.1	90.3
	QQP	90.8	86.9	90.6	83.6	88.0
	SQuADv2.0	92.8	87.0	91.4	85.8	89.2
	SQuADv1.1	92.9	89.5	92.7	85.3	90.1
	HellaSwag	92.6	87.5	91.4	86.6	89.5
	CCG	87.6	78.5	87.6	75.7	82.4
	Cosmos QA	91.8	86.9	91.7	88.4	89.7
	CSQA	86.1	80.8	87.9	81.6	84.1

Table 13: Full BUCC Results

	af	ar	bg	bn	de	el	es	et	eu	fa	fi	fr	he	hi	hu	id	it	ja	jv	
	XLM-R	30.5	20.4	39.0	13.3	63.9	18.9	48.0	25.8	19.9	42.0	41.5	48.1	28.0	38.3	42.5	47.0	42.3	41.8	10.2
Without MLM	ANLI <sup>+</sup>	78.8	74.0	88.0	72.3	97.4	82.4	91.2	70.9	53.3	91.5	88.6	89.8	82.1	92.8	86.2	<b>92.1</b>	82.6	88.7	31.7
	MNLI	79.6	70.7	84.8	71.2	96.6	82.5	93.1	74.3	59.2	90.0	89.0	89.6	81.8	91.7	86.0	91.7	86.3	89.5	30.7
	QQP	<b>80.4</b>	74.9	87.3	74.3	96.5	84.1	<b>93.8</b>	74.7	60.2	91.0	90.3	89.9	<b>86.0</b>	93.3	88.4	<b>92.1</b>	86.3	89.9	35.6
	SQuADv2.0	73.7	67.7	84.2	63.2	96.0	74.3	89.2	70.5	54.0	87.9	85.5	87.1	77.1	88.0	83.5	89.5	80.2	86.4	32.2
	SQuADv1.1	76.9	68.9	85.7	65.7	96.4	76.3	89.5	76.9	58.4	88.0	88.5	88.5	77.3	89.9	84.0	90.4	83.0	88.7	30.2
	HellaSwag	78.9	<b>75.4</b>	<b>89.9</b>	<b>75.4</b>	<b>97.7</b>	<b>84.8</b>	93.1	<b>79.8</b>	64.8	<b>91.8</b>	<b>92.0</b>	<b>92.2</b>	84.9	<b>93.4</b>	<b>89.5</b>	<b>92.1</b>	<b>86.7</b>	<b>91.6</b>	<b>37.1</b>
	CCG	71.9	59.1	82.1	62.5	95.5	74.4	87.0	67.3	49.0	84.7	82.6	84.4	77.2	85.4	80.7	87.2	79.1	78.7	24.9
	Cosmos QA	78.6	70.6	86.6	71.0	96.4	80.5	91.8	77.6	60.7	89.8	91.3	89.4	83.0	91.5	87.7	91.4	83.7	88.2	37.1
	CSQA	79.5	74.5	87.7	74.0	96.9	83.6	92.9	79.1	65.8	90.0	<b>92.0</b>	90.7	83.1	92.2	88.4	91.8	85.4	88.9	33.7
	Multi-task	81.2	71.9	88.0	73.6	97.1	82.9	92.6	73.1	58.6	90.4	89.6	89.6	84.1	92.6	87.2	92.6	83.9	91.0	34.1
With MLM	ANLI <sup>+</sup>	78.6	65.2	86.6	67.8	97.0	78.2	90.2	79.1	59.3	89.3	89.1	90.4	78.7	89.3	86.5	91.0	84.6	87.0	26.3
	MNLI	77.3	65.2	83.8	64.9	97.2	76.1	92.1	77.7	57.3	88.1	88.8	87.5	81.0	89.0	87.1	90.5	82.6	85.6	27.3
	QQP	74.4	61.3	83.7	64.6	96.2	75.7	88.1	76.7	59.4	86.3	87.0	86.9	76.6	85.9	84.2	89.8	79.8	84.0	28.8
	SQuADv2.0	70.8	57.6	80.9	52.7	96.6	63.4	84.5	71.5	47.4	85.4	86.9	85.1	71.9	85.2	83.9	90.4	78.1	83.2	16.1
	SQuADv1.1	79.2	67.7	86.5	71.4	96.7	80.4	91.6	83.1	<b>66.3</b>	90.8	91.1	89.8	77.5	92.3	87.4	91.8	84.6	87.4	26.3
	HellaSwag	57.1	45.2	69.4	40.4	89.7	57.8	73.4	64.0	42.2	77.1	76.4	76.5	62.6	75.1	76.2	82.5	69.7	77.5	22.0
	CCG	71.9	52.3	80.4	51.0	95.0	72.6	86.0	73.5	51.0	83.3	84.1	81.8	71.3	79.1	81.6	87.2	78.7	76.2	12.7
	Cosmos QA	69.7	63.7	84.0	58.8	95.1	74.2	84.6	76.5	58.6	85.7	85.2	84.5	76.2	87.1	84.7	88.5	81.4	85.5	24.9
	CSQA	54.3	45.3	63.6	33.5	87.0	50.5	70.0	58.8	35.7	74.1	71.0	70.7	58.2	70.2	72.5	80.4	64.2	75.5	16.6
		ka	kk	ko	ml	mr	nl	pt	ru	sw	ta	te	th	tl	tr	ur	vi	zh	Avg	
	XLM-R	11.8	17.4	35.5	19.4	15.2	52.6	47.2	42.1	7.9	9.1	19.7	27.4	10.3	37.8	22.5	38.3	41.2	31.0	-
Without MLM	ANLI <sup>+</sup>	76.9	67.3	84.6	90.8	80.5	<b>93.6</b>	91.0	90.5	30.8	76.5	85.5	<b>91.2</b>	59.9	87.9	79.7	94.6	<b>93.0</b>	80.8	-
	MNLI	77.9	67.7	84.3	89.8	80.4	92.5	91.3	89.2	32.8	70.0	78.2	86.7	60.9	88.8	74.5	92.5	91.2	80.2	-
	QQP	78.7	69.4	86.4	<b>92.9</b>	93.3	<b>92.5</b>	91.6	35.1	<b>81.4</b>	<b>90.6</b>	90.0	64.6	<b>91.4</b>	81.7	95.0	92.3	82.7	-	
	SQuADv2.0	67.0	63.0	80.8	82.8	71.6	89.7	90.4	86.9	27.7	60.9	74.4	80.7	54.2	85.9	70.6	92.5	89.3	76.1	-
	SQuADv1.1	70.9	63.7	83.3	87.3	74.7	91.7	90.2	89.1	31.5	60.6	77.8	82.3	59.3	88.3	68.3	92.8	90.8	77.9	-
	HellaSwag	<b>80.8</b>	<b>72.0</b>	<b>86.5</b>	92.1	81.1	93.2	91.9	<b>92.0</b>	35.1	79.2	87.2	89.6	64.5	90.6	<b>82.4</b>	<b>95.1</b>	92.6	<b>83.3</b>	-
	CCG	65.1	56.9	76.8	82.5	70.3	88.9	88.8	84.5	24.9	60.3	65.4	72.8	53.3	82.6	64.7	89.7	84.8	72.9	-
	Cosmos QA	75.7	69.9	83.6	90.1	78.7	92.0	91.3	89.7	34.1	72.3	84.6	89.1	59.7	89.6	79.8	93.3	90.9	80.9	-
	CSQA	<b>80.8</b>	70.3	85.5	91.7	82.7	93.3	91.4	90.4	<b>35.9</b>	73.3	84.6	89.4	<b>65.4</b>	90.2	77.1	94.8	92.9	82.2	-
	Multi-task	78.7	68.2	85.0	91.4	80.4	92.1	92.0	90.2	34.4	68.7	83.8	89.1	62.3	88.9	77.6	95.0	92.8	81.2	-
With MLM	ANLI <sup>+</sup>	70.6	64.7	83.6	88.9	75.6	92.0	91.0	88.1	29.0	70.0	76.9	84.7	51.6	88.0	71.7	93.6	91.6	78.5	-
	MNLI	67.7	63.3	81.8	84.3	75.0	90.8	90.5	87.8	29.7	62.2	73.5	85.2	53.4	87.6	71.2	93.3	88.5	77.4	-
	QQP	66.0	64.2	80.2	82.0	70.6	89.4	89.8	86.7	30.5	60.9	76.1	83.6	52.3	84.9	72.7	90.5	88.0	76.0	-
	SQuADv2.0	53.8	54.8	77.5	72.5	61.5	90.0	87.0	87.2	20.3	41.7	51.7	80.5	38.0	81.8	63.3	90.6	89.1	70.4	-
	SQuADv1.1	73.2	66.8	83.9	89.8	78.9	93.0	90.4	89.7	33.8	76.2	85.0	90.0	54.5	90.0	78.6	93.6	90.9	80.6	-
	HellaSwag	38.5	43.1	70.5	63.2	39.7	79.1	78.4	80.0	19.2	30.9	55.6	66.6	33.1	71.5	49.8	80.4	77.7	61.4	-
	CCG	58.3	51.3	74.6	76.3	58.4	89.0	86.9	82.9	23.3	46.9	60.3	72.6	40.9	82.5	55.8	87.9	80.3	69.4	-
	Cosmos QA	63.3	56.0	80.7	79.0	63.1	89.4	87.2	86.1	26.2	55.7	71.8	80.5	44.6	83.0	63.7	91.0	85.1	73.8	-
	CSQA	33.4	36.2	65.9	47.0	30.9	76.6	74.7	75.5	19.0	28.3	49.6	64.1	26.0	64.1	53.0	78.4	75.1	56.9	-

Table 14: Full Tatoeba Results

	MNLI	QQP	HellaSwag
en	87.1	88.0	71.6
Translated to de	82.2	84.6	55.1
Translated to ru	70.1	83.8	27.4
Translated to sw	70.8	79.3	25.1

Table 15: Intermediate task performance on trained and evaluated on translated data. We report the median result for English (original) task data.

# STIL - Simultaneous Slot Filling, Translation, Intent Classification, and Language Identification: Initial Results using mBART on MultiATIS++

Jack G. M. FitzGerald

Amazon Alexa AI

Seattle, WA

jgmf@amazon.com

## Abstract

Slot-filling, Translation, Intent classification, and Language identification, or STIL, is a newly-proposed task for multilingual Natural Language Understanding (NLU). By performing simultaneous slot filling and translation into a single output language (English in this case), some portion of downstream system components can be monolingual, reducing development and maintenance cost. Results are given using the multilingual BART model (Liu et al., 2020) fine-tuned on 7 languages using the MultiATIS++ dataset. When no translation is performed, mBART’s performance is comparable to the current state of the art system (Cross-Lingual BERT by Xu et al. (2020)) for the languages tested, with better average intent classification accuracy (96.07% versus 95.50%) but worse average slot F1 (89.87% versus 90.81%). When simultaneous translation is performed, average intent classification accuracy degrades by only 1.7% relative and average slot F1 degrades by only 1.2% relative.

## 1 Introduction

Multilingual Natural Language Understanding (NLU), also called cross-lingual NLU, is a technique by which an NLU-based system can scale to multiple languages. A single model is trained on more than one language, and it can accept input from more than one language during inference. In most recent high-performing systems, a model is first pre-trained using unlabeled data for all supported languages and then fine tuned for a specific task using a small set of labeled data (Conneau and Lample, 2019; Pires et al., 2019).

Two typical tasks for goal-based systems, such as virtual assistants and chatbots, are intent classification and slot filling (Gupta et al., 2006). Though intent classification creates a language agnostic output (the intent of the user), slot filling does not.

Input	从盐湖城到加州奥克兰的航班
Traditional Output	intent: flight slots: (盐湖城, fromloc.cityname), ... (奥克兰, toloc.cityname), ... (加州, toloc.statename)
STIL Output	intent: flight slots: (salt lake city, fromloc.cityname), ... (oakland, toloc.cityname), ... (california, toloc.statename) lang: zh

Table 1: Today’s slot filling systems do not translate the slot content, as shown in “Traditional Ouput.” With a STIL model, the slot content is translated and language identification is performed.

Instead, a slot-filling model outputs the labels for each of input tokens from the user. Suppose the slot-filling model can handle  $L$  languages. Downstream components must therefore handle all  $L$  languages for the full system to be multilingual across  $L$  languages. Machine translation could be performed before the slot filling model at system runtime, though the latency would be fully additive, and some amount of information useful to the slot-filling model may be lost. Similarly, translation could occur after the slot-filling model at runtime, but slot alignment between the source and target language is a non-trivial task (Jain et al., 2019; Xu et al., 2020). Instead, the goal of this work was to build a single model that can simultaneously translate the input, output slotted text in a single language (English), classify the intent, and classify the input language (See Table 1). The STIL task is defined such that the input language tag is not given to the model as input. Thus, language identification is necessary so that the system can communicate back to the user in the correct language.

Contributions of this work include (1) the introduction of a new task for multilingual NLU, namely simultaneous Slot filling, Translation, Intent clas-

Example Input	Example Output
flüge von salt lake city nach oakland kalifornien	salt <B-fromloc.city_name> lake <I-fromloc.city_name> city <I-fromloc.city_name> oakland <B-toloc.city_name> california <B-toloc.state_name> <intent-flight> <lang-de>
从盐湖城到加州奥克兰 的航班	salt <B-fromloc.city_name> lake <I-fromloc.city_name> city <I-fromloc.city_name> oakland <B-toloc.city_name> california <B-toloc.state_name> <intent-flight> <lang-zh>

Table 2: Two text-to-text STIL examples. In all STIL cases, the output is in English. Each token is followed by its BIO-tagged slot label. The sequence of tokens and slots are followed by the intent and then the language.

sification, and Language identification (STIL); (2) both non-translated and STIL results using the mBART model (Liu et al., 2020) trained using a fully text-to-text data format; and (3) public release of source code used in this study, with a goal toward reproducibility and future work on the STIL task<sup>1</sup>.

## 2 Dataset

The Airline Travel Information System (ATIS) dataset is a classic benchmark for goal-oriented NLU (Price, 1990; Tur et al., 2010). It contains utterances focused on airline travel, such as *how much is the cheapest flight from Boston to New York tomorrow morning?* The dataset is annotated with 17 intents, though the distribution is skewed, with 70% of intents being the *flight* intent. Slots are labeled using the Beginning Inside Outside (BIO) format. ATIS was localized to Turkish and Hindi in 2018, forming MultiATIS (Upadhyay et al., 2018), and then to Spanish, Portuguese, German, French, Chinese, and Japanese in 2020, forming MultiATIS++ (Xu et al., 2020).

In this work, Portuguese was excluded due to a lack of Portuguese pretraining in the publicly available mBART model, and Japanese was excluded due to a current lack of alignment between Japanese and English samples in MultiATIS++. Hindi and Turkish data were taken from MultiATIS, and the training data were upsampled by 3x for Hindi and 7x for Turkish. Prior to any upsampling, there were 4,488 training samples for English, Spanish, German, French, and Chinese. The test sets contained 893 samples for all languages except Turkish, which had 715 samples.

For English, Spanish, German, French, and Chinese, validation sets of 490 samples were used in all cases. Given the smaller data quantities for Hindi and Turkish, two training and validation set configurations were considered. The first configuration

matched that of Xu et al. (2020), using training sets of 1,495 for Hindi and 626 for Turkish along with validation sets of 160 for Hindi and 60 for Turkish. In the second configuration, no validation sets were made for Hindi and Turkish (though there were still validation sets for the other languages), and the training sets of 1,600 Hindi samples and 638 samples from MultiATIS were used.

Two output formats are considered, being (1) the non-translated, traditional case, in which translation of slot content is not performed, and (2) the translated, STIL case, in which translation of slot content is performed. In both cases, the tokens, the labels, the intent, and the detected language are all output from the model as a single ordered text sequence, as shown in Table 2.

## 3 Related Work

Previous approaches for intent classification and slot filling have used either (1) separate models for slot filling, including support vector machines (Moschitti et al., 2007), conditional random fields (Xu and Sarikaya, 2014), and recurrent neural networks of various types (Kurata et al., 2016) or (2) joint models that diverge into separate decoders or layers for intent classification and slot filling (Xu and Sarikaya, 2013; Guo et al., 2014; Liu and Lane, 2016; Hakkani-Tür et al., 2016) or that share hidden states (Wang et al., 2018). In this work, a fully text-to-text approach similar to that of the T5 model was used, such that the model would have maximum information sharing across the four STIL sub-tasks.

Encoder-decoder models, first introduced in 2014 (Sutskever et al., 2014), are a mainstay of neural machine translation. The original transformer model included both an encoder and a decoder (Vaswani et al., 2017). Since then, much of the work on transformers focuses on models with only an encoder pretrained with autoencoding techniques (e.g. BERT by Devlin et al. (2018)) or auto-regressive models with only a decoder (e.g.

<sup>1</sup><https://github.com/jgmfitz/stil-mbart-multiatispp-aacl2020>

GPT by Radford (2018)). In this work, it was assumed that encoder-decoder models, such as BART (Lewis et al., 2019) and T5 (Raffel et al., 2019), are the best architectural candidates given the translation component of the STIL task, as well as past state of the art advancement by encoder-decoder models on ATIS, cited above. Rigorous architectural comparisons are left to future work.

## 4 The Model

### 4.1 The Pretrained mBART Model

The multilingual BART (mBART) model architecture was used (Liu et al., 2020), as well as the pretrained mBART.cc25 model described in the same paper. The model consists of 12 encoder layers, 12 decoder layers, a hidden layer size of 1,024, and 16 attention heads, yielding a parameter count of 680M. The mBART.cc25 model was trained on 25 languages for 500k steps using a 1.4 TB corpus of scraped website data taken from Common Crawl (Wenzek et al., 2019). The model was trained to reconstruct masked tokens and to rearrange scrambled sentences. SentencePiece tokenization (Kudo and Richardson, 2018) was used for mBART.cc25 with a sub-word vocabulary size of 250k.

### 4.2 This Work

The same vocabulary as that of the pretrained model was used for this work, and SentencePiece tokenization was performed on the full sequence, including the slot tags, intent tags, and language tags. For all mBART experiments and datasets, data from all languages were shuffled together. The fairseq library was used for all experimentation (Ott et al., 2019).

Training was performed on 8 Nvidia V100 GPUs (16 GB) using a batch size of 32, layer normalization for both the encoder and the decoder (Xu et al., 2019); label smoothed cross entropy with  $\epsilon = 0.2$  (Szegedy et al., 2016); the ADAM optimizer with  $\beta_1 = 0.9$  and  $\beta_2 = 0.999$  (Kingma and Ba, 2014); an initial learning rate of  $3 \times 10^{-5}$  with polynomial decay over 20,000 updates after 1 epoch of warmup; attention dropout of 0.1 and dropout of 0.2 elsewhere; and FP16 type for weights. Each model was trained for 19 epochs, which took 5-6 hours.

## 5 Results and Discussion

Results from the models are given in Table 3. Statistical significance was evaluated using the Wilson

method (Wilson, 1927) with 95% confidence.

### 5.1 Comparing to Xu et al. (2020)

Examining the first training configuration (1,496 samples for Hindi and 626 for Turkish), the non-translated mBART’s macro-averaged intent classification (96.07%) outperforms Cross-Lingual BERT by Xu et al. (2020) (95.50%), but slot F1 is worse (89.87% for non-translated mBART and 90.81% for Cross-Lingual BERT). The differences are statistically significant in both cases.

### 5.2 With and Without Translation

When translation is performed (the STIL task), intent classification accuracy degrades by 1.7% relative from 96.07% to 94.40%, and slot F1 degrades by 1.2% relative from 89.87% to 88.79%. The greatest degradation occurred for utterances involving flight number, airfare, and airport name (in that order).

### 5.3 Additional Hindi and Turkish Training Data

Adding 105 more Hindi and 12 more Turkish training examples results in improved performance for the translated, STIL mBART model. Macro-averaged intent classification improves from 94.40% to 95.94%, and slot F1 improves from 88.79% to 90.10%, both of which are statistically significant. By adding these 117 samples, the STIL mBART model matches the performance (within confidence intervals) of the non-translated mBART model. This finding suggests that the STIL models may require more training data than traditional, non-translated slot filling models.

Additionally, by adding more Hindi and Turkish data, both the intent accuracy and the slot filling F1 improves for every individual language of the translated, STIL models, suggesting that some portion of the internal, learned representation is language agnostic.

Finally, the results suggest that there is a training-size-dependent performance advantage in using a single output language, as contrasted with the non-translated mBART model, for which the intent classification accuracy and slot F1 does not improve (with statistical significance) when using the additional Hindi and Turkish training samples.

### 5.4 Language Identification

Language identification F1 is above 99.7% for all languages, with perfect performance in many cases.

<b>Intent accuracy</b>	<b>en</b>	<b>es</b>	<b>de</b>	<b>zh</b>	<b>fr</b>	<b>hi</b>	<b>tr</b>	<b>Mac Avg</b>
Cross-Lingual BERT (Xu et al., 2020)	97.20	96.77	96.86	95.54	97.24	92.70 tr=1495	92.20 tr=626	95.50
Seq2Seq-Ptr (Rongali et al., 2020)	97.42							
Stack Propagation (Qin et al., 2019)	97.5							
Joint BERT + CRF (Chen et al., 2019)	97.9							
Non-translated mBART, with hi-tr val	96.98	96.98	97.09	96.08	97.65	95.07 tr=1495	92.73 tr=626	96.07
Translated/STIL mBART, with hi-tr val	95.86	94.62	95.63	93.84	95.97	93.84 tr=1495	91.05 tr=626	94.40
Non-translated mBART, no hi-tr val	97.09	97.20	97.20	96.30	97.42	94.74 tr=1600	94.27 tr=638	96.32
Translated/STIL mBART, no hi-tr val	96.98	96.53	96.64	96.42	97.31	94.85 tr=1600	92.87 tr=638	95.94
<b>Slot F1</b>	<b>en</b>	<b>es</b>	<b>de</b>	<b>zh</b>	<b>fr</b>	<b>hi</b>	<b>tr</b>	<b>Mac Avg</b>
Bi-RNN (Upadhyay et al., 2018)	95.2					80.6 tr=600	78.9 tr=600	84.90
Cross-Lingual BERT (Xu et al., 2020)	95.90	87.95	95.00	93.67	90.39	86.73 tr=1495	86.04 tr=626	90.81
Stack Propagation (Qin et al., 2019)	96.1							
Joint BERT (Chen et al., 2019)	96.1							
Non-translated mBART, with hi-tr val	95.03	86.76	94.42	92.13	89.31	86.91 tr=1495	84.53 tr=626	89.87
Translated/STIL mBART, with hi-tr val	93.81	90.38	91.41	85.93	91.24	83.98 tr=1495	84.79 tr=626	88.79
Non-translated mBART, no hi-tr val	95.00	86.87	94.14	92.22	89.32	87.42 tr=1600	84.33 tr=638	89.90
Translated/STIL mBART, no hi-tr val	94.66	91.55	92.61	87.73	92.15	86.74 tr=1600	85.23 tr=638	90.10
<b>Language Identification F1</b>	<b>en</b>	<b>es</b>	<b>de</b>	<b>zh</b>	<b>fr</b>	<b>hi</b>	<b>tr</b>	<b>Mac Avg</b>
Translated/STIL mBART, with hi-tr val	100.00	98.87	100.00	100.00	98.95	100.00	99.93	99.68
Translated/STIL mBART, no hi-tr val	99.78	99.83	100.00	100.00	99.72	100.00	99.86	99.88

Table 3: Results are shown for intent accuracy, slot F1 score, and language identification F1 score. For English, Spanish, German, Chinese, and French in all of the models shown above (including other work), training sets were between 4,478 and 4,488 samples, and validation sets were between 490 and 500 samples. In this work, two training set sizes were used for Hindi and Turkish, denoted by “tr=” and “with hi-tr val[itation set]” or “no hi-tr val[itation set]”. Across all work shown above, the tests sets contained 893 samples for all languages except Turkish, for which the test set was 715 samples.

Perfect performance on Chinese and Hindi is unsurprising given their unique scripts versus the other languages tested.

## 6 Conclusion

This preliminary work demonstrates that a single NLU model can perform simultaneous slot filling, translation, intent classification, and language identification across 7 languages using MultiATIS++. Such an NLU model would negate the need for multiple-language support in some portion of downstream system components. Performance is not irreconcilably worse than traditional slot-filling models, and performance is statistically equivalent with a small amount of additional training data.

Looking forward, a more challenging dataset is needed to further develop the translation compo-

nent of the STIL task. The English MultiATIS++ test set only contains 455 unique entity-slot pairs. An ideal future dataset would include freeform and varied content, such as text messages, song titles, or open-domain questions. Until then, work remains to achieve parity with English-only ATIS models.

## Acknowledgments

The author would like to thank Saleh Soltan, Gokhan Tur, Saab Mansour, and Batool Haider for reviewing this work and providing valuable feedback.

## References

- Qian Chen, Zhu Zhuo, and Wen Wang. 2019. Bert for joint intent classification and slot filling. *ArXiv*, abs/1902.10909.

- Alexis Conneau and Guillaume Lample. 2019. [Cross-lingual language model pretraining](#). In *Advances in Neural Information Processing Systems 32*, pages 7059–7069. Curran Associates, Inc.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. [Bert: Pre-training of deep bidirectional transformers for language understanding](#).
- Daniel (Zhaohan) Guo, Gokhan Tur, Scott Wen-tau Yih, and Geoffrey Zweig. 2014. [Joint semantic utterance classification and slot filling with recursive neural networks](#). In *2014 IEEE Spoken Language Technology Workshop (SLT 2014)*. IEEE - Institute of Electrical and Electronics Engineers.
- N. Gupta, G. Tur, D. Hakkani-Tur, S. Bangalore, G. Riccardi, and M. Gilbert. 2006. The at t spoken language understanding system. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(1):213–222.
- Dilek Hakkani-Tür, Gokhan Tur, Asli Celikyilmaz, Yun-Nung Vivian Chen, Jianfeng Gao, Li Deng, and Ye-Yi Wang. 2016. [Multi-domain joint semantic frame parsing using bi-directional rnn-lstm](#). In *Proceedings of The 17th Annual Meeting of the International Speech Communication Association (INTERSPEECH 2016)*. ISCA.
- Alankar Jain, Bhargavi Paranjape, and Zachary C. Lipton. 2019. [Entity projection via machine translation for cross-lingual NER](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1083–1092, Hong Kong, China. Association for Computational Linguistics.
- Diederik P. Kingma and Jimmy Ba. 2014. [Adam: A method for stochastic optimization](#).
- Taku Kudo and John Richardson. 2018. [Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing](#). *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*.
- Gakuto Kurata, Bing Xiang, Bowen Zhou, and Mo Yu. 2016. [Leveraging sentence-level information with encoder lstm for semantic slot filling](#). *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.
- Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. [Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension](#).
- Bing Liu and Ian Lane. 2016. [Attention-based recurrent neural network models for joint intent detection and slot filling](#). *Interspeech 2016*.
- Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. [Multilingual denoising pre-training for neural machine translation](#).
- Alessandro Moschitti, Giuseppe Riccardi, and Christian Raymond. 2007. Spoken language understanding with kernels for syntactic/semantic structures. *2007 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 183–188.
- Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. [fairseq: A fast, extensible toolkit for sequence modeling](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.
- Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. [How multilingual is multilingual BERT?](#) In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.
- P. J. Price. 1990. [Evaluation of spoken language systems: the ATIS domain](#). In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*.
- Libo Qin, Wanxiang Che, Yangming Li, Haoyang Wen, and Ting Liu. 2019. [A stack-propagation framework with token-level intent detection for spoken language understanding](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2078–2087, Hong Kong, China. Association for Computational Linguistics.
- Alec Radford. 2018. Improving language understanding by generative pre-training.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. [Exploring the limits of transfer learning with a unified text-to-text transformer](#).
- Subendhu Rongali, Luca Soldaini, Emilio Monti, and Wael Hamza. 2020. [Don't parse, generate! a sequence to sequence architecture for task-oriented semantic parsing](#). In *Proceedings of The Web Conference 2020, WWW '20*, page 2962–2968, New York, NY, USA. Association for Computing Machinery.
- Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2, NIPS'14*, page 3104–3112, Cambridge, MA, USA. MIT Press.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826.

Gokhan Tur, Dilek Z. Hakkani-Tür, and Larry Heck. 2010. What is left to be understood in atis? In *2010 IEEE Spoken Language Technology Workshop*, pages 19–24.

Shyam Upadhyay, Manaal Faruqui, Gokhan Tur, Dilek Hakkani-Tür, and Larry Heck. 2018. (almost) zero-shot cross-lingual spoken language understanding. In *Proceedings of the IEEE ICASSP*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS’17*, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Yu Wang, Yilin Shen, and Hongxia Jin. 2018. A bi-model based RNN semantic frame parsing model for intent detection and slot filling. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 309–314, New Orleans, Louisiana. Association for Computational Linguistics.

Guillaume Wenzek, Marie-Anne Lachaux, Alexis Conneau, Vishrav Chaudhary, Francisco Guzmán, Armand Joulin, and Edouard Grave. 2019. Ccnet: Extracting high quality monolingual datasets from web crawl data.

Edwin B. Wilson. 1927. Probable inference, the law of succession, and statistical inference. *Journal of the American Statistical Association*, 22(158):209–212.

Jingjing Xu, Xu Sun, Zhiyuan Zhang, Guangxiang Zhao, and Junyang Lin. 2019. Understanding and improving layer normalization. In *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc.

P. Xu and R. Sarikaya. 2013. Convolutional neural network based triangular crf for joint intent detection and slot filling. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 78–83.

Puyang Xu and Ruhi Sarikaya. 2014. Targeted feature dropout for robust slot filling in natural language understanding. ISCA - International Speech Communication Association.

Weijia Xu, Batool Haider, and Saab Mansour. 2020. End-to-end slot alignment and recognition for cross-lingual nlu.

# SimulMT to SimulST: Adapting Simultaneous Text Translation to End-to-End Simultaneous Speech Translation

Xutai Ma

Johns Hopkins University

xutai\_ma@jhu.edu

Juan Pino

Facebook AI

juancarabina@fb.com

Philipp Koehn

Johns Hopkins University

phi@jhu.edu

## Abstract

Simultaneous text translation and end-to-end speech translation have recently made great progress but little work has combined these tasks together. We investigate how to adapt simultaneous text translation methods such as wait- $k$  and monotonic multihead attention to end-to-end simultaneous speech translation by introducing a pre-decision module. A detailed analysis is provided on the latency-quality trade-offs of combining fixed and flexible pre-decision with fixed and flexible policies. We also design a novel computation-aware latency metric, adapted from Average Lagging (AL) (Ma et al., 2019).

## 1 Introduction

Simultaneous speech translation (SimulST) generates a translation from an input speech utterance before the end of the utterance has been heard. SimulST systems aim at generating translations with maximum quality and minimum latency, targeting applications such as video caption translations and real-time language interpreter. While great progress has recently been achieved on both end-to-end speech translation (Ansari et al., 2020) and simultaneous text translation (SimulMT) (Grissom II et al., 2014; Gu et al., 2017; Luo et al., 2017; Lawson et al., 2018; Alinejad et al., 2018; Zheng et al., 2019b,a; Ma et al., 2020; Arivazhagan et al., 2019, 2020), little work has combined the two tasks together (Ren et al., 2020).

End-to-end SimulST models feature a smaller model size, greater inference speed and fewer compounding errors compared to their cascade counterpart, which perform streaming speech recognition followed by simultaneous machine translation. In addition, it has been demonstrated that end-to-end SimulST systems can have lower latency than cascade systems (Ren et al., 2020).

<sup>1</sup>The code is available at <https://github.com/pytorch/fairseq>

In this paper, we study how to adapt methods developed for SimulMT to end-to-end SimulST. To this end, we introduce the concept of pre-decision module. Such module guides how to group encoder states into meaningful units prior to making a READ/WRITE decision. A detailed analysis of the latency-quality trade-offs when combining a fixed or flexible pre-decision module with a fixed or flexible policy is provided. We also introduce a novel computation-aware latency metric, adapted from Average Lagging (AL) (Ma et al., 2019).

## 2 Task formalization

A SimulST model takes as input a sequence of acoustic features  $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_{|\mathbf{X}|}]$  extracted from speech samples every  $T_s$  ms, and generates a sequence of text tokens  $\mathbf{Y} = [y_1, \dots, y_{|\mathbf{Y}|}]$  in a target language. Additionally, it is able to generate  $y_i$  with only partial input  $\mathbf{X}_{1:n(y_i)} = [\mathbf{x}_1, \dots, \mathbf{x}_{n(y_i)}]$ , where  $n(y_i) \leq |\mathbf{X}|$  is the number of frames needed to generate the  $i$ -th target token  $y_i$ . Note that  $n$  is a monotonic function, i.e.  $n(y_{i-1}) \leq n(y_i)$ .

A SimulST model is evaluated with respect to quality, using BLEU (Papineni et al., 2002), and latency. We introduce two latency evaluation methods for SimulST that are adapted from SimulMT. We first define two types of delays to generate the word  $y_i$ , a computation-aware (CA) and a non computation-aware (NCA) delay. The CA delay of  $y_i$ ,  $d_{\text{CA}}(y_i)$ , is defined as the time that elapses (speech duration) from the beginning of the process to the prediction of  $y_i$ , while the NCA delay for  $y_i$ ,  $d_{\text{NCA}}(y_i)$  is defined by  $d_{\text{NCA}}(y_i) = T_s \cdot n(y_i)$ . Note that  $d_{\text{NCA}}$  is an ideal case for  $d_{\text{CA}}$  where the computational time for the model is ignored. Both delays are measured in milliseconds. Two types of latency measurement,  $L_{\text{CA}}$  and  $L_{\text{NCA}}$ , are calculated accordingly:  $L = \mathcal{C}(\mathbf{D})$  where  $\mathcal{C}$  is a latency metric and  $\mathbf{D} = [d(y_1), \dots, d(y_{|\mathbf{Y}|})]$ .

To better evaluate the latency for SimulST, we introduce a modification to AL. We assume an oracle system that can perform perfect simultaneous translation for both latency and quality, while in Ma et al. (2019) the oracle is ideal only from the latency perspective. We evaluate the lagging based on time rather than steps. The modified AL metric is defined in Eq. (1):

$$AL = \frac{1}{\tau(|X|)} \sum_{i=1}^{\tau(|X|)} d(y_i) - \frac{|X|}{|Y^*|} \cdot T_s \cdot (i-1) \quad (1)$$

where  $|Y^*$  is the length of the reference translation,  $\tau(|X|)$  is the index of the first target token generated when the model read the full input. There are two benefits from this modification. The first is that latency is measured using time instead of steps, which makes it agnostic to preprocessing and segmentation. The second is that it is more robust and can prevent an extremely low and trivial value when the prediction is significantly shorter than the reference.

### 3 Method

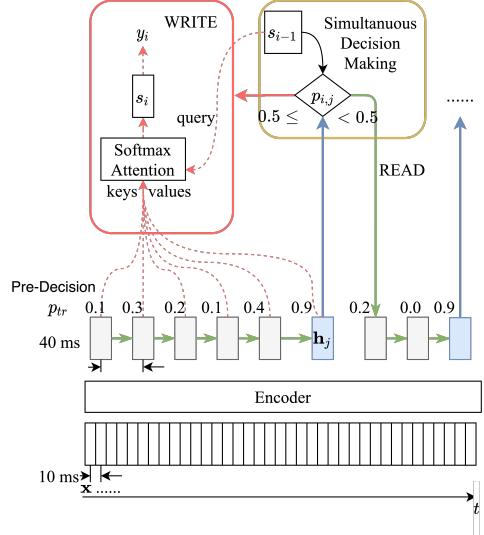
#### 3.1 Model Architecture

End-to-end ST models directly map a source speech utterance into a sequence of target tokens. We use the S-Transformer architecture proposed by (Di Gangi et al., 2019b), which achieves competitive performance on the MuST-C dataset (Di Gangi et al., 2019a). In the encoder, a two-dimensional attention is applied after the CNN layers and a distance penalty is introduced to bias the attention towards short-range dependencies.

We investigate two types of simultaneous translation mechanisms, flexible and fixed policy. In particular, we investigate monotonic multihead attention (Ma et al., 2020), which is an instance of flexible policy and the prefix-to-prefix model (Ma et al., 2019), an instance of fixed policy, designated by wait- $k$  from now on.

**Monotonic Multihead Attention (MMA)** (Ma et al., 2020) extends monotonic attention (Raffel et al., 2017; Arivazhagan et al., 2019) to Transformer-based models. Each head in each layer has an independent step probability  $p_{ij}$  for the  $i$ th target and  $j$ th source step, and then uses a closed form expected attention for training. A weighted average and variance loss were proposed to control the behavior of the attention heads and thus the trade-offs between quality and latency.

**Wait- $k$**  (Ma et al., 2019) is a fixed policy that waits for  $k$  source tokens, and then reads and writes alternatively. Wait- $k$  can be a special case of Monotonic Infinite-Lookback Attention (MILk) (Arivazhagan et al., 2019) or MMA where the step-wise probability  $p_{ij} = 0$  if  $j - i < k$  else  $p_{ij} = 1$ .



**Figure 1:** Simul-ST architecture with pre-decision module. Blue states in the figure indicate the point Simul-SST model triggers the simultaneous making process

#### 3.2 Pre-Decision Module

In SimulMT, READ or WRITE decisions are made at the token (word or BPE) level. However, with speech input, it is unclear when to make such decisions. For example, one could choose to read or write after each frame or after generating each encoder state. Meanwhile, a frame typically only covers 10ms of the input while an encoder state generally covers 40ms of the input (assuming a subsampling factor of 4), while the average length of a word in our dataset is 270ms. Intuitively, a policy like wait- $k$  will not have enough information to write a token after reading a frame or generating an encoder state. In principle, a flexible or model-based policy such as MMA should be able to handle granular while MMA is more robust to input. Our analysis will show, however, that the granularity of the input, it also performs poorly when the input is too fine-grained.

In order to overcome these issues, we introduce the notion of pre-decision module, which groups frames or encoder states, prior to making a decision. A pre-decision module generates a series of trigger probabilities  $p_{tr}$  on each encoder states to indicate whether a simultaneous decision should be

made. If  $p_{tr} > 0.5$ , the model triggers the simultaneous decision making, otherwise keeps reading new frames. We propose two types of pre-decision module.

**Fixed Pre-Dcision** A straightforward policy for a fixed pre-decision module is to trigger simultaneous decision making every fixed number of frames. Let  $\Delta t$  be the time corresponding to this fixed number of frames, with  $\Delta t$  a multiple of  $T_s$ , and  $r_e = \text{int}(|\mathbf{X}|/|\mathbf{H}|)$ .  $p_{tr}$  at encoder step  $j$  is defined in Eq. (2):

$$p_{tr}(j) = \begin{cases} 1 & \text{if } \text{mod}(j \cdot r_e \cdot T_s, \Delta t) = 0, \\ 0 & \text{Otherwise.} \end{cases} \quad (2)$$

**Flexible Pre-Dcision** We use an oracle flexible pre-decision module that uses the source boundaries either at the word or phoneme level. Let  $\mathbf{A}$  be the alignment between encoder states and source labels (word or phoneme).  $\mathbf{A}(h_i)$  represents the token that  $h_i$  aligns to. The trigger probability can then be defined in Eq. (3):

$$p_{tr}(j) = \begin{cases} 0 & \text{if } \mathbf{A}(h_j) = \mathbf{A}(h_{j-1}) \\ 1 & \text{Otherwise.} \end{cases} \quad (3)$$

## 4 Experiments

We conduct experiments on the English-German portion of the MuST-C dataset (Di Gangi et al., 2019a), where source audio, source transcript and target translation are available. We train on 408 hours of speech and 234k sentences of text data. We use Kaldi (Povey et al., 2011) to extract 80 dimensional log-mel filter bank features, computed with a 25ms window size and a 10ms window shift. For text, we use SentencePiece (Kudo and Richardson, 2018) to generate a unigram vocabulary of size 10,000. We use Gentle<sup>2</sup> to generate the alignment between source text and speech as the label to generate the oracle flexible pre-decision module. Translation quality is evaluated with case-sensitive detokenized BLEU with SACREBLEU (Post, 2018). The latency is evaluated with our proposed modification of AL (Ma et al., 2019). All results are reported on the MuST-C dev set.

All speech translation models are first pre-trained on the ASR task where the target vocabulary is character-based, in order to initialize the

<sup>2</sup><https://lowerquality.com/gentle/>

encoder. We follow the same hyperparameter settings from (Di Gangi et al., 2019b). We follow the latency regularization method introduced by (Ma et al., 2020; Arivazhagan et al., 2019). The objective function to optimize is

$$L = -\log(P(\mathbf{Y}|\mathbf{X})) + \lambda \max(\mathcal{C}(\mathbf{D}), 0) \quad (4)$$

Where  $\mathcal{C}$  is a latency metric (AL in this case) and  $\mathbf{D}$  is described in Section 2. Only samples with  $AL > 0$  are regularized to avoid overfitting. For the models with monotonic multihead attention, we first train a model without latency with  $\lambda_{\text{latency}} = 0$ . After the model converges,  $\lambda_{\text{latency}}$  is set to a desired value and the model is continue trained until convergence.

The latency-quality trade-offs of the 4 types of model from the combination of fixed or flexible pre-decision with fixed or flexible policy are presented in Fig. 2. The non computation-aware delays are used to calculate the latency metric in order to evaluate those trade-offs from a purely algorithmic perspective.

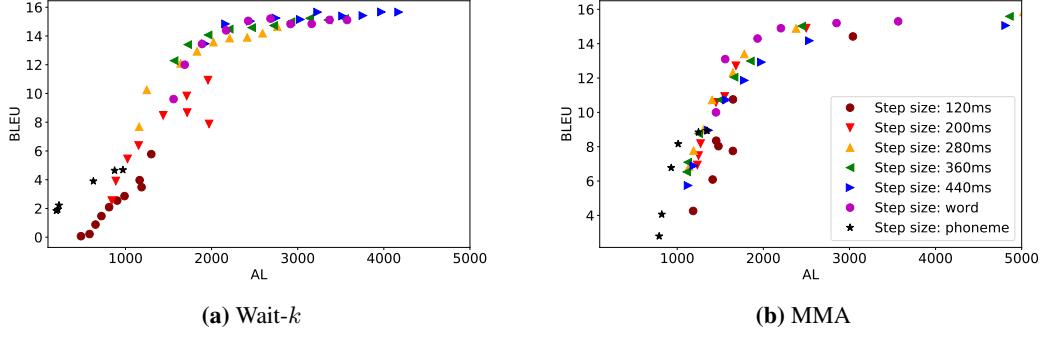
**Fixed Pre-Dcision + Fixed Policy** <sup>3</sup> (Fig. 2a). As expected, both quality and latency increase with step size and lagging. In addition, the latency-quality trade-offs are highly dependent on the step size of the pre-decision module. For example, with step size 120ms, the performance is very poor even with large  $k$  because of very limited information being read before writing a target token. Large step sizes improve the quality but introduce a lower bound on the latency. Note that step size 280ms, which provides an effective latency-quality trade-off compared to other step sizes, also matches the average word length of 271ms. This motivates the study of a flexible pre-decision module based on word boundaries.

**Fixed Pre-Dcision + Flexible Policy** <sup>4</sup> (Fig. 2b) Similar to wait- $k$ , MMA obtains very poor performance with a small step size of 120ms. For other step sizes, MMA obtains similar latency-quality trade-offs, demonstrating some form of robustness to the step size.

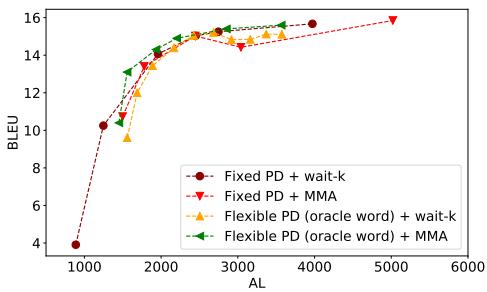
**Flexible Pre-Dcision** Curve  $\star$  and  $\bullet$  in figure Fig. 2 show latency-quality trade-offs when the pre-decision module is determined by oracle word or phoneme boundaries. Note that a SimulST model

<sup>3</sup> $k = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10$

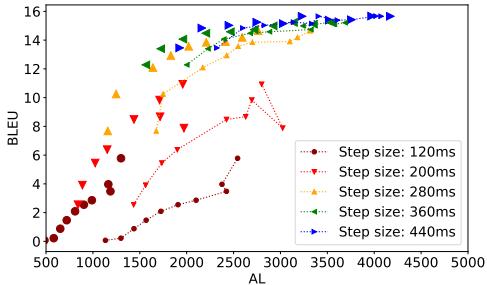
<sup>4</sup> $\lambda = 0.001, 0.004, 0.01, 0.02, 0.04, 0.06, 0.08, 0.1$



**Figure 2:** Latency-Quality trade-off curves. The unit of AL is millisecond



**Figure 3:** Comparison of best models in four settings



**Figure 4:** Computation-aware latency for fixed pre-decision + wait-*k* policy. Points on dotted lines are computation-aware, without lines are non-computation-aware

would not normally have access to this information and that the purpose of this experiment is to guide future design of a flexible pre-decision model. First, as previously observed, the granularity of the pre-decision greatly influences the latency-quality trade-offs. Models using phoneme boundaries obtain very poor translation quality because those boundaries are too granular, with an average phoneme duration of 77ms. In addition, comparing MMA and wait-*k* with phoneme boundaries, MMA is found to be more robust to the granularity of the pre-decision.

**Best Curves** The best settings for each approach

are compared in Fig. 3. For fixed pre-decision, we choose the setting that has the best quality for each latency bucket of 500ms, while for the flexible pre-decision we use oracle word boundaries. For both wait-*k* and MMA, the flexible pre-decision module outperforms the fixed pre-decision module. This is expected since the flexible pre-decision module uses oracle information in the form of pre-computed word boundaries but provides a direction for future research. The best latency-quality trade-offs are obtained with MMA and flexible pre-decision from word boundaries.

#### 4.1 Computation Aware Latency

We also consider the computation-aware latency described in Section 2, shown in Fig. 4. The focus is on fixed pre-decision approaches in order to understand the relation between the granularity of the pre-decision and the computation time. Fig. 4 shows that as the step size increases, the difference between the NCA and the CA latency shrinks. This is because with larger step sizes, there is less overhead of recomputing the bidirectional encoder states<sup>5</sup>. We recommend future work on SimulST to make use of CA latency as it reflects a more realistic evaluation, especially in low-latency regimes, and is able to distinguish streaming capable systems.

### 5 Conclusion

We investigated how to adapt SimulMT methods to end-to-end SimulST by introducing the concept of pre-decision module. We also adapted Average Lagging to be computation-aware. The effects of combining a fixed or flexible pre-decision module

<sup>5</sup>This is a common practice in SimulMT where the input length is significantly shorter than in SimulST (Arivazhagan et al., 2019; Ma et al., 2019; Arivazhagan et al., 2020)

with a fixed or flexible policy were carefully analyzed. Future work includes building an incremental encoder to reduce the CA latency and design a learnable pre-decision module.

## References

- Ashkan Alinejad, Maryam Siahbani, and Anoop Sarkar. 2018. Prediction improves simultaneous neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3022–3027.
- Ebrahim Ansari, amittai axelrod, Nguyen Bach, Ondřej Bojar, Roldano Cattoni, Fahim Dalvi, Nadir Durranı, Marcello Federico, Christian Federmann, Jiatao Gu, Fei Huang, Kevin Knight, Xutai Ma, Ajay Nagesh, Matteo Negri, Jan Niehues, Juan Pino, Elizabeth Salesky, Xing Shi, Sebastian Stüker, Marco Turchi, Alexander Waibel, and Changhan Wang. 2020. **FINDINGS OF THE IWSLT 2020 EVALUATION CAMPAIGN**. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 1–34, Online. Association for Computational Linguistics.
- Naveen Arivazhagan, Colin Cherry, Wolfgang Macherey, Chung-Cheng Chiu, Semih Yavuz, Ruoming Pang, Wei Li, and Colin Raffel. 2019. Monotonic infinite lookback attention for simultaneous machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1313–1323, Florence, Italy. Association for Computational Linguistics.
- Naveen Arivazhagan, Colin Cherry, Wolfgang Macherey, and George Foster. 2020. Re-translation versus streaming for simultaneous translation. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 220–227, Online. Association for Computational Linguistics.
- Mattia A. Di Gangi, Roldano Cattoni, Luisa Bentivogli, Matteo Negri, and Marco Turchi. 2019a. **MuST-C: a Multilingual Speech Translation Corpus**. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2012–2017, Minneapolis, Minnesota. Association for Computational Linguistics.
- Mattia Antonino Di Gangi, Matteo Negri, Roldano Cattoni, Roberto Densi, and Marco Turchi. 2019b. Enhancing transformer for end-to-end speech-to-text translation. In *Proceedings of Machine Translation Summit XVII Volume 1: Research Track*, pages 21–31, Dublin, Ireland. European Association for Machine Translation.
- Alvin Grissom II, He He, Jordan Boyd-Graber, John Morgan, and Hal Daumé III. 2014. Don’t until the final verb wait: Reinforcement learning for simultaneous machine translation. In *Proceedings of the 2014 Conference on empirical methods in natural language processing (EMNLP)*, pages 1342–1352.
- Jiatao Gu, Graham Neubig, Kyunghyun Cho, and Victor OK Li. 2017. Learning to translate in real-time with neural machine translation. In *15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017*, pages 1053–1062. Association for Computational Linguistics (ACL).
- Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71.
- Dieterich Lawson, Chung-Cheng Chiu, George Tucker, Colin Raffel, Kevin Swersky, and Navdeep Jaitly. 2018. Learning hard alignments with variational inference. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5799–5803. IEEE.
- Yuping Luo, Chung-Cheng Chiu, Navdeep Jaitly, and Ilya Sutskever. 2017. Learning online alignments with continuous rewards policy gradient. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2801–2805. IEEE.
- Mingbo Ma, Liang Huang, Hao Xiong, Renjie Zheng, Kaibo Liu, Baigong Zheng, Chuanqiang Zhang, Zhongjun He, Hairong Liu, Xing Li, Hua Wu, and Haifeng Wang. 2019. **STACL: Simultaneous translation with implicit anticipation and controllable latency using prefix-to-prefix framework**. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3025–3036, Florence, Italy. Association for Computational Linguistics.
- Xutai Ma, Juan Pino, James Cross, Liezl Puzon, and Jiatao Gu. 2020. **Monotonic multihead attention**. In *International Conference on Learning Representations*.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.
- Matt Post. 2018. **A call for clarity in reporting BLEU scores**. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.
- Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko

Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. 2011. The kaldi speech recognition toolkit. In *IEEE 2011 workshop on automatic speech recognition and understanding*, CONF. IEEE Signal Processing Society.

Colin Raffel, Minh-Thang Luong, Peter J Liu, Ron J Weiss, and Douglas Eck. 2017. Online and linear-time attention by enforcing monotonic alignments. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2837–2846. JMLR.org.

Yi Ren, Jinglin Liu, Xu Tan, Chen Zhang, Tao QIN, Zhou Zhao, and Tie-Yan Liu. 2020. **SimulSpeech: End-to-end simultaneous speech to text translation**. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3787–3796, Online. Association for Computational Linguistics.

Baigong Zheng, Renjie Zheng, Mingbo Ma, and Liang Huang. 2019a. **Simpler and faster learning of adaptive policies for simultaneous translation**. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1349–1354, Hong Kong, China. Association for Computational Linguistics.

Baigong Zheng, Renjie Zheng, Mingbo Ma, and Liang Huang. 2019b. **Simultaneous translation with flexible policy via restricted imitation learning**. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5816–5822, Florence, Italy. Association for Computational Linguistics.

# Cue Me In: Content-Inducing Approaches to Interactive Story Generation

Faeze Brahman<sup>†</sup>, Alexandru Petrusca<sup>†</sup>, and Snigdha Chaturvedi<sup>‡</sup>

<sup>†</sup>Department of Computer Science and Engineering, University of California, Santa Cruz

<sup>‡</sup>Department of Computer Science, University of North Carolina at Chapel Hill

{fbrahman, apetrusca}@ucsc.edu snigdha@cs.unc.edu

## Abstract

Automatically generating stories is a challenging problem that requires producing causally related and logical sequences of events about a topic. Previous approaches in this domain have focused largely on one-shot generation, where a language model outputs a complete story based on limited initial input from a user. Here, we instead focus on the task of interactive story generation, where the user provides the model mid-level sentence abstractions in the form of cue phrases *during* the generation process. This provides an interface for human users to guide the story generation. We present two content-inducing approaches to effectively incorporate this additional information. Experimental results from both automatic and human evaluations show that these methods produce more topically coherent and personalized stories compared to baseline methods.

## 1 Introduction

Automatic story generation requires composing a coherent and fluent passage of text about a sequence of events. Prior studies on story generation mostly focused on symbolic planning (Lebowitz, 1987; Pérez y Pérez and Sharples, 2001; Porteous and Cavazza, 2009; Riedl and Young, 2010) or case-based reasoning (Gervás et al., 2005) that heavily relied on manual knowledge engineering.

Recent state-of-the-art methods for story generation (Martin et al., 2018; Clark et al., 2018a) are based on sequence-to-sequence models (Sutskever et al., 2014) that generate a story in one go. In this setting, the user has little control over the generated story.

On the other hand, when humans write, they incrementally edit and refine the text they produce. Motivated by this, rather than generating the entire story at once, we explore the problem of interactive story generation. In this setup, a user can provide

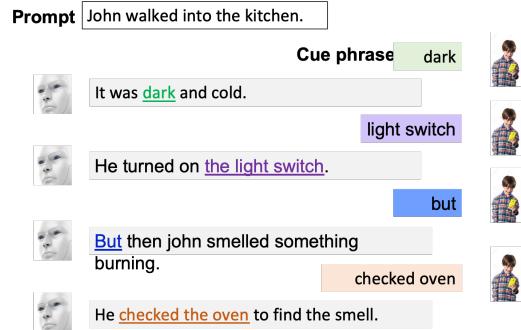


Figure 1: Interactive story generation: the user inputs the first sentence of the story (prompt), and provides guiding cue phrases as the system generates the story one sentence at a time.

the model mid-level sentence abstractions in the form of *cue phrases* as the story is being generated. Cue phrases enable the user to inform the system of what they want to happen next in the story and have more control over what is being generated. To achieve our goal, this paper primarily focuses on approaches for smoothly and effectively incorporating user-provided cues. The schematic in Fig. 1 illustrates this scenario: the system generates the story one sentence at a time, and the user guides the content of the next sentence using cue phrases. We note that the generated sentences need to fit the context, and also be semantically related to the provided cue phrase.

A fundamental advantage of using this framework as opposed to a fully automated one is that it can provide an interactive interface for human users to incrementally supervise the generation by giving signals to the model throughout the story generation process. This human-computer collaboration can result in generating richer and personalized stories. In particular, this field of research can be used in addressing the literacy needs of learners with disabilities and enabling children to explore

creative writing at an early age by crafting their own stories.

In this paper, we present two content-inducing approaches based on the Transformer Network (Vaswani et al., 2017) for interactively incorporating external knowledge when automatically generating stories. Here, our external knowledge is in the form of cue phrases provided by the user to enable interaction, but can readily be replaced with knowledge accessible through other means<sup>1</sup>. Specifically, our models fuse information from the story context and cue phrases through a hierarchical attention mechanism. The first approach, *Cued Writer*, employs two independent encoders (for incorporating context and cue phrases) and an additional attention component to capture the semantic agreement between the cue phrase and output sentence. The second approach, *Relevance Cued Writer*, additionally measures the relatedness between the context and cue phrase through a context-cue multi-head unit. In both cases, we introduce different attention units in a single end-to-end neural network.

Our automatic and human evaluations demonstrate that the presented models outperform strong baselines and can successfully incorporate cues in generated stories. This capability is one step closer to an interactive setup, and unlike one-shot generation, it lets users have more control over the generation. Our contributions are twofold:

- Two novel content-inducing approaches to incorporate additional information, in this case cue phrases, into the generation phase.
- Experiments demonstrating utility of content-inducing approaches using automatic and human evaluations.

## 2 Related Work

Automatic story generation is a longstanding problem in AI, with early work dating back to the 1970s based on symbolic planning (Lebowitz, 1987; Pérez y Pérez and Sharples, 2001; Porteous and Cavazza, 2009; Riedl and Young, 2010) and case-based reasoning using ontologies (Gervás et al., 2005). Li et al. (2013) extended prior works toward learning domain models (via corpus and/or crowdsourcing) to support open story generation about any topic.

<sup>1</sup>For example, the user-provided cues can be replaced by the outputs of an automatic planner. Our models are flexible enough to work in other setups.

With the advent of deep learning there has been a major shift towards using seq2seq models (Sutskever et al., 2014; Bahdanau et al., 2015) for various text generation tasks, including storytelling (Roemmele, 2016; Jain et al., 2017; Hu et al., 2020). However, these models often fail to ensure coherence in the generated story. To address this problem, Clark et al. (2018a) incorporated entities given their vector representations, which get updated as the story unfolds. Similarly, Liu et al. (2020) proposed a character-centric story generation by learning character embeddings directly from the corpus. Fan et al. (2018) followed a two-step process to first generate the premise and then condition on that to generate the story. Yu et al. (2020) proposed a multi-pass CVAE to improve wording diversity and content consistency.

Previous work has explored the potential of creative writing with a machine in the loop. Clark et al. (2018b) found that people generally enjoy collaborating with a machine. Traditional methods proposed to write stories collaboratively using a case-based reasoning architecture (Swanson and Gordon, 2012). Recent work (Roemmele and Gordon, 2015) extended this to find relevant suggestions for the next sentence in a story from a large corpus. Other methods proposed GUI and tools to facilitate co-creative narrative generation (Manjavacas et al., 2017; Kapadia et al., 2015). Unlike us, these approaches explore the value of and tools for interaction rather than designing methods for incorporating user input into the model.

Another line of research decomposes story generation into two steps: story plot planning and plot-to-surface generation. Previous work produces story-plans based on sequences of events (Martin et al., 2018; Tambwekar et al., 2019; Ammanabrolu et al., 2020), critical phrases (Xu et al., 2018) or both events and entities (Fan et al., 2019). Yao et al. (2019) model the story-plan as a sequence of keywords. They proposed *Static* and *Dynamic* paradigms that generate a story based on these story-plans. Goldfarb-Tarrant et al. (2019) adopted the *static* model proposed in Yao et al. (2019) to supervise story-writing.

A major focus of these works is on generating a coherent plan for generating the story. In contrast, our contribution is complementary since we do not focus on *planning* but on *generation*. We present approaches to effectively incorporate external knowledge in the form of cue-phrases during

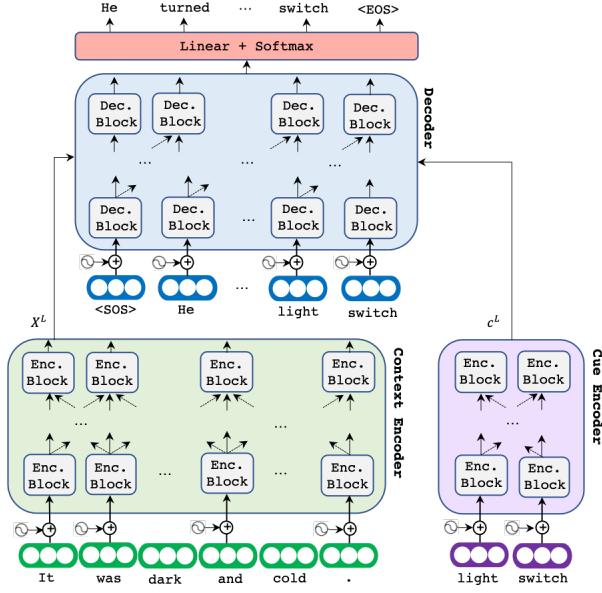


Figure 2: Overall model architecture for *Cued Writer* and *Relevance Cued Writer*.

generation, and conduct extensive experiments to compare our models with those of Yao et al. (2019) by modifying them to work in our setup.

### 3 Interactive Story Generation

We design models to generate a story one sentence at a time. Given the generated context so far (as a sequence of tokens)  $X = \{x_1, \dots, x_T\}$ , and the cue phrase for the next sentence  $c = \{c_1, \dots, c_K\}$ , our models generate the tokens of the next sentence of the story  $Y = \{y_1, \dots, y_M\}$ . We train the models by minimizing the cross-entropy loss:

$$L_\theta = - \sum_{i=1}^M \log P(y_i | X, c, \theta) \quad (1)$$

Here,  $\theta$  refers to model parameters. Note that when generating the  $n$ -th sentence, the model takes the first  $n - 1$  sentences in the story as the context along with the cue phrase.

In the rest of this section, we describe our two novel content-inducing approaches for addressing the interactive story generation task: the *Cued Writer*, and the *Relevance Cued Writer*. These models share an overall encoder-decoder based architecture shown in Fig. 2. They adopt a dual encoding approach where two separate but architecturally similar encoders are used for encoding the context (Context Encoder represented in the green box) and the cue phrase (Cue Encoder represented in the purple box). Both these encoders

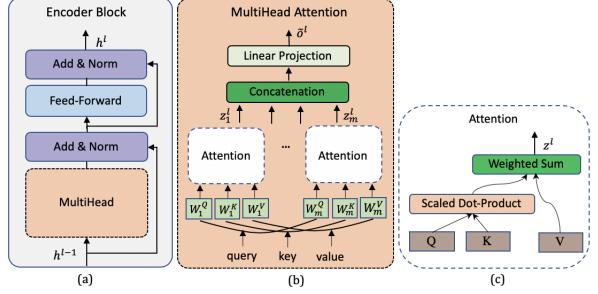


Figure 3: (a) Encoder Block consists of MultiHead and FFN. (b) MultiHead Attention. (c) Attention Module.

advise the Decoder (represented in the blue box), which in turn generates the next sentence. The two proposed models use the same encoding mechanism (described in § 3.1) and differ only in their decoders (described in § 3.2).

### 3.1 Encoder

Our models use the Transformer encoder introduced in Vaswani et al. (2017). Here, we provide a generic description of the encoder architecture followed by the inputs to this architecture for the Context and Cue Encoders in our models.

Each encoder layer  $l$  contains architecturally identical Encoder Blocks, referred to as ENCBLOCK (with unique trainable parameters). Fig. 3(a) shows an Encoder Block which consists of a Multi-Head attention and an FFN that applies the following operations:

$$\tilde{o}^l = \text{MULTIHEAD}(h^{l-1}) \quad (2a)$$

$$o^l = \text{LAYERNORM}(\tilde{o}^l + h^{l-1}) \quad (2b)$$

$$\tilde{h}^l = \text{FFN}(o^l) \quad (2c)$$

$$h^l = \text{LAYERNORM}(\tilde{h}^l + o^l) \quad (2d)$$

Where MULTIHEAD represents Multi-Head Attention (described below), FFN is a feed-forward neural network with ReLU activation (LeCun et al., 2015), and LAYERNORM is a layer normalization (Ba et al., 2016). In the rest of the paper, LAYERNORM (also shown as Add & Norm in figures) is always applied after MULTIHEAD and FFN, but we do not explicitly mention that in text or equations for simplicity.

**Multi-Head Attention** The multi-head attention, shown in Fig. 3(b), is similar to that used in Vaswani et al. (2017). It is made of multiple Attention heads, shown in Fig. 3(c). The Attention head has three types of inputs: the query sequence,

$Q \in R^{n_q \times d_k}$ , the key sequence,  $K \in R^{n_k \times d_k}$ , and the value sequence,  $V \in R^{n_v \times d_k}$ . The attention module takes each token in the query sequence and attends to tokens in the key sequence using a scaled dot product. The score for each token in the key sequence is then multiplied by the corresponding value vector to form a weighted sum:

$$\text{ATTN}(Q, K, V) = \text{softmax} \left( \frac{QK^T}{\sqrt{d_k}} \right) V \quad (3)$$

For each head, all  $Q$ ,  $K$ , and  $V$  are passed through a head-specific projection prior to the attention being computed. The output of a single head is:

$$H_i = \text{ATTN}(QW_i^Q, KW_i^K, VW_i^V) \quad (4)$$

Where  $W$ s are head-specific projections. Attention heads  $H_i$  are then concatenated:

$$\text{MULTIHEAD}(Q, K, V) = [H_1; \dots; H_m]W^O \quad (5)$$

Where  $W^O$  is an output projection. In the encoder, all query, key, and value come from the previous layer and thus:

$$\text{MULTIHEAD}(h^{l-1}) = \text{MULTIHEAD}(h^{l-1}, h^{l-1}, h^{l-1}) \quad (6)$$

**Encoder Input** The Encoder Blocks described above form the constituent units of the Context and Cue Encoders, which process the context and cue phrase respectively. Each token in the context,  $x_i$ , and cue phrase,  $c_i$ , is assigned two kinds of embeddings: *token embeddings* indicating the meaning and *position embeddings* indicating the position of each token within the sequence. These two are summed to obtain individual input vectors,  $X^0$ , and  $c^0$ , which are then fed to the first layer of Context and Cue encoders, respectively. Thereafter, new representations are constructed through layers of encoder blocks:

$$X^{l+1} = \text{ENCBLOCK}(X^l, X^l, X^l) \quad (7a)$$

$$c^{l+1} = \text{ENCBLOCK}(c^l, c^l, c^l) \quad (7b)$$

where  $l \in [0, L - 1]$  denotes different layers. In Eqn. 7a and 7b, the output of the previous layer's Encoder Block is used as  $Q$ ,  $K$ , and  $V$  input for the multi-head attention of the next block.

### 3.2 Content-Inducing Decoders

We now describe the decoders for our models.

**Cued Writer** The main intuition behind our first model, *Cued Writer*, is that since cue phrases

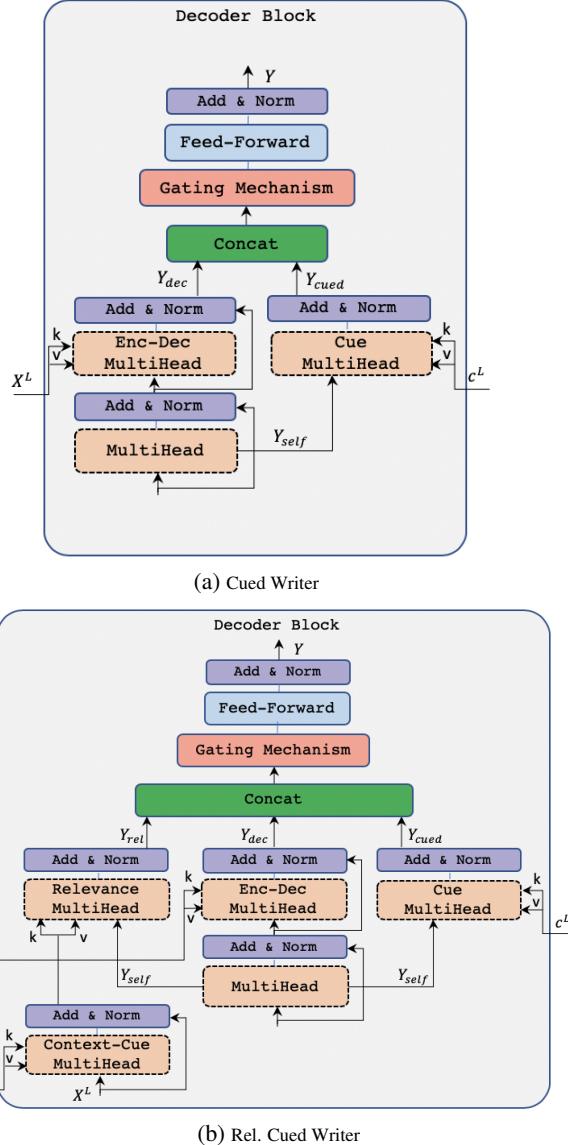


Figure 4: Decoder architectures.  $X^L$  and  $c^L$  are the outputs of the top-layers of the Context and Cue encoders respectively, and  $K$  and  $V$  are the corresponding keys and values.

indicate users' expectations of what they want to see in the next sentence of the story, they should be used by the model *at the time of generation*, i.e., in the decoder. Below, we describe the decoder used by the *Cued Writer*.

After processing the two types of inputs in the Context and Cue Encoders, the model includes their final encoded representations ( $X^L$  and  $c^L$ ) in the decoder. The decoder consists of  $L$  layers with architecturally identical Decoder Blocks. Each Decoder Block contains Enc-Dec MultiHead and the Cue MultiHead units (see Fig. 4(a)), which let the decoder to focus on the relevant parts of the context and the cue phrase, respectively.

Given  $Y^0$  as the word-level embedding represen-

tation for the output sentence, our Decoder Block is formulated as:

$$Y_{self}^{l+1} = \text{MULTIH}(Y^l, Y^l, Y^l) \quad (8a)$$

$$Y_{dec}^{l+1} = \text{MULTIH}(Y_{self}^{l+1}, X^L, X^L) \quad (8b)$$

$$Y_{cued}^{l+1} = \text{MULTIH}(Y_{self}^{l+1}, c^L, c^L) \quad (8c)$$

Eqn. 8a is standard self-attention, which measures the intra-sentence agreement for the output sentence and corresponds to the MultiHead unit in Fig. 4(a). Eqn. 8b, describing the Enc-Dec MultiHead unit, measures the agreement between context and output sentence, where queries come from the decoder Multi-Head unit ( $Y_{self}$ ), and the keys and values come from the top layer of the context encoder ( $X^L$ ). Similarly, Eqn. 8c captures the agreement between output sentence and cue phrase through Cue MultiHead unit. Here, keys and values come from the top layer of the Cue encoder ( $c^L$ ).

Lastly, we adapt a gating mechanism (Sriram et al., 2018) to integrate the semantic representations from both  $Y_{dec}$  and  $Y_{cued}$  and pass the resulting output to FFN function:

$$g^{l+1} = \sigma(W_1[Y_{dec}^{l+1}; Y_{cued}^{l+1}]) \quad (9a)$$

$$Y_{int}^{l+1} = W_2(g^{l+1} \circ [Y_{dec}^{l+1}; Y_{cued}^{l+1}]) \quad (9b)$$

$$Y^{l+1} = \text{FFN}(Y_{int}^{l+1}) \quad (9c)$$

the representation from  $Y_{dec}$  and  $Y_{cued}$  are concatenated to learn gates,  $g$ . The gated hidden layers are combined by concatenation and followed by a linear projection with the weight matrix  $W_2$ .

**Relevance Cued Writer** The decoder of *Cued Writer* described above captures the relatedness of the context and the cue phrase to the generated sentence but does not study the relatedness or relevance of the cue phrase to the context. We incorporate this relevance in the decoder of our next model, *Relevance Cued Writer*. Its Decoder Block (shown in Fig. 4(b)) is similar to that of *Cued Writer* except for two additional units: the Context-Cue and Relevance MultiHead units. The intuition behind the Context-Cue MultiHead unit (Eqn. 10a) is to characterize the relevance between the context and the cue phrase, so as to highlight the effect of words in the cue phrase that are more relevant to the context thereby promoting topicality and fluency. This relevance is then provided to the decoder using the Relevance MultiHead unit (Eqn. 10b):

$$X_{rel}^{l+1} = \text{MULTIH}(X^L, c^L, c^L) \quad (10a)$$

$$Y_{rel}^{l+1} = \text{MULTIH}(Y_{self}^{l+1}, X_{rel}^{l+1}, X_{rel}^{l+1}) \quad (10b)$$

We fuse the information from all three sources using a gating mechanism and pass the result to FFN:

$$g^{l+1} = \sigma(W_1[Y_{dec}^{l+1}; Y_{cued}^{l+1}; Y_{rel}^{l+1}]) \quad (11a)$$

$$Y_{int}^{l+1} = W_2(g^{l+1} \circ [Y_{dec}^{l+1}; Y_{cued}^{l+1}; Y_{rel}^{l+1}]) \quad (11b)$$

$$Y^{l+1} = \text{FFN}(Y_{int}^{l+1}) \quad (11c)$$

Finally, for both models, a linear transformation and a softmax function (shown in Fig. 2) is applied to convert the output produced by the stack of decoders to predicted next-token probabilities:

$$P(y_i|y_{<i}, X, c, \theta) = \text{softmax}(Y_i^L W_y) \quad (12)$$

where  $P(y_i|y_{<i}, X, c, \theta)$  is the likelihood of generating  $y_i$  given the preceding text ( $y_{<i}$ ), context and cue, and  $W_y$  is the token embedding matrix.

## 4 Empirical Evaluation

### 4.1 Dataset

We used the ROCStories corpus (Mostafazadeh et al., 2016) for experiments. It contains 98,161 five-sentence long stories with a rich set of causal/temporal sequences of events. We held out 10% of stories for validation and 10% for test set.

### 4.2 Baselines

**SEQ2SEQ** Our first baseline is based on a LSTM sentence-to-sentence generator with attention (Bahdanau et al., 2015). In order to incorporate user-provided cue phrases, we concatenate context and cue phrase with a delimiter token (<\$>) before passing it to the encoder.

**DYNAMIC** This is the Dynamic model proposed by Yao et al. (2019) modified to work in our setting. For a fair comparison, instead of generating a plan, we provide the model with cue phrases and generate the story one sentence at a time.

**STATIC** The STATIC model (Yao et al., 2019) gets all cue phrases at once to generate the entire story<sup>2</sup>. By design, it has additional access to all, including future, cue phrases. Our models and other baselines do not have this information.

**VANILLA** To verify the effectiveness of our content-inducing approaches, we use a Vanilla Transformer as another baseline and concatenate context and cue phrase using a delimiter token.

---

<sup>2</sup>We used the implementation available at: <https://bitbucket.org/VioletPeng/language-model/>

Models	PPL ( $\downarrow$ )	BLEU-1 ( $\uparrow$ )	BLEU-2 ( $\uparrow$ )	BLEU-3 ( $\uparrow$ )	GM ( $\uparrow$ )	Repetition-4 ( $\downarrow$ )
DYNAMIC (Yao et al., 2019)	29.49	30.05	9.16	4.59	0.73	44.36
STATIC (Yao et al., 2019)	20.81	33.25	9.64	4.77	0.75	26.26
SEQ2SEQ	20.97	33.91	10.01	3.09	0.82	33.23
VANILLA	15.78	40.30	16.09	7.19	0.89	20.87
Cued Writer	14.80	41.50	16.72	7.25	0.92	<b>15.08</b>
Rel. Cued Writer	<b>14.66</b>	<b>42.65</b>	<b>17.33</b>	<b>7.59</b>	<b>0.94</b>	16.23

Table 1: Automatic evaluation results. Our models outperform all baselines across all metrics ( $p < 0.05$ ).

### 4.3 Training details

Following previous work (Vaswani et al., 2017), we initialize context encoders and decoders with 6 layers (512 dimensional states and 8 attention heads). Our models contain 3-layer encoders for encoding cue phrases (all other specifications are the same). For the position-wise feed-forward networks, we use 2048 dimensional inner states. We use the Adam optimizer (Kingma and Ba, 2015) with a learning rate of 0.0001 and residual, embedding, and attention dropouts with a rate of 0.1 for regularization. Models are implemented in PyTorch, trained for 30 epochs with early stopping on validation loss.

**Cue-phrases for Training and Automatic Evaluation:** For training all models, we need cue phrases, which are, in principle, to be entered by a user. However, to scale model training, we automatically extracted cue phrases from the target sentences in the training set using the previously proposed RAKE algorithm (Rose et al., 2010). It is important to note that cue phrases can represent a variety of information, and many other methods can be used to extract them for training purposes. For example, topic words, distinctive entities or noun phrases in the sentence, the headword in the dependency parse of the sentence, etc.

Our automatic evaluations were done on a large-scale, and so we followed a similar approach for extracting cue-phrases.

**Cue-phrases for Human Evaluation:** In the interest of evaluating the interactive nature of our models, cue-phrases were provided manually during our interactive evaluations<sup>3</sup>.

**General Statistics on Cue-phrases:** Automatically extracted cue phrases has the vocabulary size of 22,097, and 6,189 on the train and test set, respectively with the average 10% coverage over the entire target sentence. Cue-phrases are typically 1-2 words. Comparing user-provided vs automati-

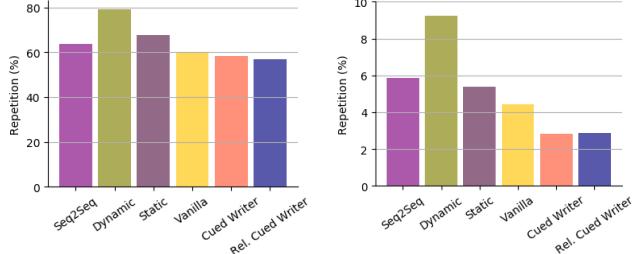


Figure 5: Inter-story (left) and Intra-story (right) repetition scores. The proposed models have better scores.

cally extracted cue-phrases, the average length of user-provided cue-phrases in interactive evaluation is 1.56, with a vocabulary size of 206, whereas these numbers are 1.59 and 214 for their corresponding automatically extracted cue phrases.

### 4.4 Automatic Evaluation

Following previous credible works (Martin et al., 2018; Fan et al., 2018), we compare various methods using Perplexity and BLEU (Papineni et al., 2002) on the test set. We reported BLEU- $n$  for  $n=1, 2, 3$ . From Table 1, we can see that both our models outperform DYNAMIC and STATIC by large margins on perplexity and BLEU scores. The proposed models are also superior to the SEQ2SEQ and VANILLA baseline on both measures. Comparing the last two rows of Table 1, we also see an additive gain from modeling the relevance in *Rel. Cued Writer*. All improvements are statistically significant (approximate randomization (Noreen, 1989),  $p < 0.05$ ).

To evaluate how well the story generation model incorporates the cues, we use an embedding-based greedy matching score (GM) (Liu et al., 2016). The score measures the relatedness of the generated story with cues by greedily matching them with each token in a story based on the cosine similarity of their word embeddings (Yao et al., 2019). We can see from the 5th column in Table 1 that our models generate stories that are more related to the cue phrases.

<sup>3</sup>We left the definition of cue-phrase open-ended to enable flexibility in user interaction. They are typically 1-2 words.

<b>Prompt (first sentence):</b>	Jordan was watching TV on her couch.
<b>Cue phrases:</b>	watch football - change channel - comedy show - very funny
	She was trying to watch football on TV. Then she went to change channel. Finally, she decided to watch a comedy show. She saw the comedy that was playing and didn't like.
<b>Cue phrases:</b>	soccer - cook - order pizza - tasty dinner

Her brother was playing in a soccer. She wasn't able to cook. Instead, she ordered pizza. Her brother was happy with the tasty dinner.

Table 2: Example of stories generated in interactive evaluation using two models given the same prompt and different set of cue-phrase.

Previous works have shown that neural generation models suffer from repetition issue; and so we additionally evaluate the models using repetition-4 which measures the percentage of generated stories that repeat at least one 4-gram (Shao et al., 2019) and inter- and intra-story repetition scores (Yao et al., 2019). A lower value is better for these scores. The result of repetition-4 is reported in the last column of Table 1. The proposed models significantly outperform all baselines, and among the two *Cued Writer* is better. Inter and intra repetition scores are depicted in Fig. 5. Our two proposed models are almost comparable on these metrics but they show a general superior performance compared to all baselines. In particular, *Rel. Cued Writer* achieves a significant performance increase of 16% and 46% on these scores over the stronger model of Yao et al. (2019)<sup>4</sup>.

#### 4.5 Human Evaluation

Automatic metrics cannot evaluate all aspects of open-ended text generation (Fan et al., 2018), and so we also conduct several human evaluations.

**Interactive Evaluation** In this experiment, human subjects compare our best model, *Rel. Cued Writer*, with the strongest baseline from the automatic evaluations (VANILLA) in an interactive, real-time setup.

For robust evaluation, it is essential that the users generate a wide variety of stories. Since generating different prompts (first sentence) requires creativity on the part of human judges and can be challenging, we provided participants with initial prompts that were randomly selected from the test set. For each prompt, the participants generated stories using both models by interactively provid-

<sup>4</sup>Note that the result of our SEQ2SEQ baseline is not directly comparable with that of Inc-S2S in (Yao et al., 2019), since we included cue phrases as additional input whereas Inc-S2S generate the whole story conditioned on the title.

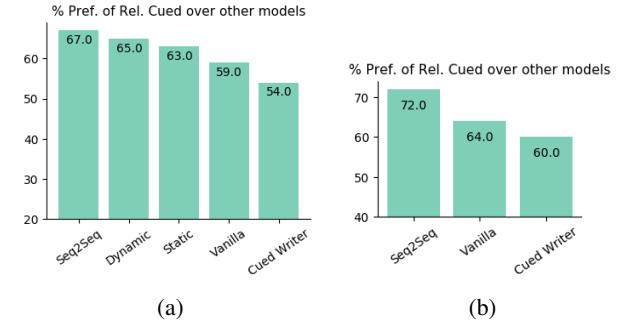


Figure 6: Human evaluations on story-level (left) and sentence-level (right). We find that human judges preferred stories generated by *Rel. Cued Writer*.

ing cue-phrases<sup>5</sup>. They were then asked to choose which story they prefer. Participants preferred stories generated by *Rel. Cued Writer* over VANILLA in 57.5% of the cases (80 stories in total,  $p \sim 0.1$ ).

Judges also rated the stories in terms of fluency and coherence on a 5-point Likert scale. *Rel. Cued Writer* achieved a higher fluency score of 4.22 compared with 3.80 achieved by VANILLA. VANILLA attained a slightly higher coherence score (3.40 vs. 3.35). On manually inspecting the generated stories, we found that our model generates longer sentences (avg. 9.18 words) with more complex language, whereas VANILLA generated relatively shorter sentences (avg. 7.46 words) which might improve coherence.

This experiment is promising but inconclusive because for the same prompt, the participants could provide different sets of cue-phrases for different models, resulting in generated stories that are too different to be comparable (Table 2 shows an example). This led us to conduct the following more controlled evaluations.

**Story-level Evaluation** In this experiment, we again make pairwise comparisons, but both models are provided the same prompts, and sets of cue phrases<sup>6</sup>. 3 judges evaluated 100 pairs of stories (in shuffled order)<sup>7</sup>.

Fig. 6(a) shows the percentage of preference for our stronger model, *Rel. Cued Writer*, over the baselines. Judges prefer our model over all other baselines. Also, judges preferred *Rel. Cued Writer* over *Cued Writer*, which demonstrates the effec-

<sup>5</sup>We included instructions and examples for participants. The order of the presentation of the models was random. The judges were self-identified native English speakers.

<sup>6</sup>Cue-phrases were extracted automatically to alleviate any biases, then curated manually.

<sup>7</sup>For all AMT evaluations, we only allow judges who have at least 500 previous jobs, and at least 97% approval rate.

<b>Prompt</b>	I was coughing very badly.		<b>Cue phrases</b>	notice blood - emergency room - waited - okay
<b>DYNAMIC</b>	I noticed that I had a headache. I went to the doctor. The doctor told me that I had to go to <b>the doctor</b> . The doctor said I was fine. <b>[waited]</b>			
<b>STATIC</b>	<b>She</b> noticed a huge blood pressure. She went to the doctor. <b>The doctor told her to stop</b> . She waited for a while. <b>[okay]</b>			
<b>SEQ2SEQ</b>	I noticed blood and <b>noticed blood</b> in the closet. <b>Now I have emergency room</b> for <b>emergency room</b> . I found it and I waited until the next day. I feel guilty and now i don't have to do it again. <b>[okay]</b>			
<b>VANILLA</b>	I noticed blood in the bathroom. I had to go to the emergency room. I waited in the bathroom for a while. Luckily <b>the doctor was okay</b> .			
<b>Cued Writer</b>	I noticed blood in my mouth. I had to go to the emergency room. I waited there a few hours. I felt okay.			
<b>Rel Cued Writer</b>	I noticed blood on my shirt. I went to the emergency room. I waited for the doctor. He came and I was okay.			
<b>Prompt</b>	Harry really wanted to see the new Star Wars movie.		<b>Cue phrases</b>	opening day - ticket - entire movie - really
<b>DYNAMIC</b>	He went to the theater to show up. He bought a ticket. He had a great time. <b>He had a great time. [opening day] [entire movie]</b>			
<b>STATIC</b>	One day, <b>she</b> was in a hurry to get to the theater. she wasn't sure if she would like to go. She was excited to try the new movie. <b>She was so excited</b> to see the movie. <b>[opening day] [ticket]</b>			
<b>SEQ2SEQ</b>	The day <b>day</b> was opening day and his family was <b>opening the opening day</b> . Harry had bought a ticket and the <b>ticket</b> wasn't very good. The entire movie was very happy. Harry became <b>very really</b> disappointed.			
<b>VANILLA</b>	On opening day, Harry was very nervous. He bought a ticket <b>to</b> the theater. He bought <b>Harry</b> ticket <b>tickets</b> to the theater. He really didn't like the movie. <b>[entire movie]</b>			
<b>Cued Writer</b>	On opening day, he went to the theater . He bought a ticket at the theater. The entire movie was great. He really was excited.			
<b>Rel Cued Writer</b>	He decided to watch it on opening day. He got to the theater and got a ticket. He watched the entire movie. He was really excited about it.			

Table 3: Sample stories generated by different models. We highlight in different color the **[missing]** cue phrase, **incoherent or unfluent**, and **repetitive** parts of each story. We see that compared to baselines, our models correctly mention cue phrases and generate better stories.

tiveness of the additional Context-Cue and Relevance Multi-Head units. All improvements are statistically significant (app. rand.,  $p < 0.05$ ).

**Sentence-level Evaluation** We also performed a more fine-grained evaluation of the models by evaluating generated sentences while the model is generating a story. The generated sentences are evaluated in light of the (incomplete) story. Specifically, we provide an (incomplete) story passage and a manually provided cue phrase to the two models to generate the next sentence. We asked human judges to identify which of the two sentences is better based on their fluency and semantic relevance to (1) the input (incomplete) story and (2) the cue phrase. We did this experiment for a set of 100 randomly selected stories (400 sentences. 3 different judges evaluated each sentence pair. Fig. 6(b) shows that the *Rel. Cued Writer* model was preferred over SEQ2SEQ and VANILLA in 72% and 64% of the cases, respectively. Comparing the two proposed models, we again see additive gain by modeling Cue-Context relevance. All improvements are statistically significant (app. rand.,  $p < 0.001$ ).

## 5 Qualitative Results and Error Analysis

Table 3 presents examples of stories generated by different models for the same prompt and cue phrases. We highlight the **[missing]** cue phrases, **incoherent or unfluent**, and **repetitive** parts of each

**off-topic:** Kelly and her friends went to a new ice-cream shop. They decided to try the new flavors. They all tried on many different restaurants. To their surprise, they thought it tasted good. They were glad to find one online.

**Not-logically-consistent:** Avery received a homework assignment due in two weeks. He immediately read it. When he turned it in, he made schedule. He completed tasks and turned it in time. When he finished early, he was disappointed.

**non-coreferent-pronouns:** Rob has never been on a roller-coaster. They go on all the way to six flags. He got on with a free ticket. Rob joined the rollercoaster. There was a long line of people in the line.

Table 4: Examples of errors made by our model.

story. Note that we did not highlight **[missing]**, if the model mentions part of the cue phrase or incorporates it semantically. As we observe, all of the baselines suffered from several issues; however, our novel content inducing approaches generate more causally related sentences, which fit the given prompt and cue phrases more naturally.

We also manually reviewed 50 stories, generated from our models and analyzed common errors. Table 4 shows sample stories that depict different types of errors including “getting off-topic”, “not-logically-connected” and “non-coreferent pronouns”. The last type of error represents the cases where the model generates pronouns that do not refer to any previously mentioned entity. The examples demonstrate that there are still many challenges in this domain.

## 6 Conclusion and Future Work

This paper explored the problem of interactive storytelling, which leverages human and computer collaboration for creative language generation. We presented two content-inducing approaches that take user-provided inputs as the story progresses and effectively incorporate them in the generated text. Experimental results show that our methods outperform competitive baselines. However, there are several other significant aspects to be considered in story generation, such as modeling of discourse relations, and representation of key narrative elements, which lie beyond the scope of this investigation. Also, while we received encouraging feedback from users on this setup during the interactive evaluation, we did not explore important questions about user interfaces, design, and human computer interaction. Future work can explore these questions and also explore other forms of natural language interaction.

## References

- Prithviraj Ammanabrolu, Ethan Tien, Wesley Cheung, Zhaochen Luo, William Ma, Lara J. Martin, and Mark O. Riedl. 2020. [Story realization: Expanding plot events into sentences](#). In *The Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 7375–7382.
- Jimmy Ba, Ryan Kiros, and Geoffrey E. Hinton. 2016. [Layer normalization](#). *CoRR*.
- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. [Neural machine translation by jointly learning to align and translate](#). In *3rd International Conference on Learning Representations*.
- Elizabeth Clark, Yangfeng Ji, and Noah A. Smith. 2018a. [Neural text generation in stories using entity representations as context](#). In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2250–2260.
- Elizabeth Clark, Anne Spencer Ross, Chenhao Tan, Yangfeng Ji, and Noah A. Smith. 2018b. [Creative writing with a machine in the loop: Case studies on slogans and stories](#). In *Proceedings of the 23rd International Conference on Intelligent User Interfaces*, pages 329–340.
- Angela Fan, Mike Lewis, and Yann Dauphin. 2018. [Hierarchical neural story generation](#). In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898.
- Angela Fan, Mike Lewis, and Yann Dauphin. 2019. [Strategies for structuring story generation](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2650–2660.
- Pablo Gervás, Belén Díaz-Agudo, Federico Peinado, and Raquel Hervás. 2005. [Story plot generation based on CBR](#). In *Applications and Innovations in Intelligent Systems XII*, 28(1):33–46.
- Seraphina Goldfarb-Tarrant, Haining Feng, and Nanyun Peng. 2019. [Plan, write, and revise: an interactive system for open-domain story generation](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Demonstrations*, pages 89–97.
- Junjie Hu, Yu Cheng, Zhe Gan, Jingjing Liu, Jianfeng Gao, and Graham Neubig. 2020. [What makes a good story? Designing composite rewards for visual storytelling](#). In *Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 7969–7976.
- Parag Jain, Priyanka Agrawal, Abhijit Mishra, Mohak Sukhwani, Anirban Laha, and Karthik Sankaranarayanan. 2017. [Story generation from sequence of independent short descriptions](#). *Workshop on Machine Learning for Creativity*.
- Mubbasis Kapadia, Jessica Falk, Fabio Zünd, Marcel Marti, Robert W. Sumner, and Markus H. Gross. 2015. [Computer-assisted authoring of interactive narratives](#). In *Proceedings of the 19th Symposium on Interactive 3D Graphics and Games*, pages 85–92.
- Diederik P. Kingma and Jimmy Ba. 2015. [Adam: A method for stochastic optimization](#). In *3rd International Conference on Learning Representations*.
- Micheal Lebowitz. 1987. Planning stories. In *Proceedings of the cognitive science society*, pages 234–242.
- Yann LeCun, Y Bengio, and Geoffrey Hinton. 2015. [Deep learning](#). *Nature*, 521:436–44.
- Boyang Li, Stephen Lee-Urban, George Johnston, and Mark Riedl. 2013. [Story generation with crowd-sourced plot graphs](#). In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*, page 598–604.
- Chia-Wei Liu, Ryan Lowe, Iulian Serban, Mike Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. [How NOT to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation](#). In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2122–2132.
- Danyang Liu, Juntao Li, Meng-Hsuan Yu, Ziming Huang, Gongshen Liu, Dongyan Zhao, and Rui Yan.

2020. A character-centric neural model for automated story generation. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 1725–1732.
- Enrique Manjavacas, Folgert Karsdorp, Ben Burtenshaw, and Mike Kestemont. 2017. Synthetic literature: Writing science fiction in a co-creative process. In *Proceedings of the Workshop on Computational Creativity in Natural Language Generation*, pages 29–37.
- Lara J. Martin, Prithviraj Ammanabrolu, Xinyu Wang, William Hancock, Shruti Singh, Brent Harrison, and Mark O. Riedl. 2018. Event representations for automated story generation with deep neural nets. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 868–875.
- Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 839–849.
- Eric W. Noreen. 1989. *Computer-Intensive Methods for Testing Hypotheses: An Introduction*. Wiley New York.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BIEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318.
- Rafael Pérez y Pérez and Mike Sharples. 2001. Mexica: A computer model of a cognitive account of creative writing. *Journal of Experimental Theoretical Artificial Intelligence*, 13(2):119–139.
- Jullie Porteous and Mike Cavazza. 2009. Controlling narrative generation with planning trajectories: the role of constraints. In *Joint International Conference on Interactive Digital Storytelling*, pages 234–245.
- Mark O. Riedl and R. Michael Young. 2010. Narrative planning: Balancing plot and character. *Journal of Artificial Intelligence Research*, 39:217–268.
- Melissa Roemmele. 2016. Writing stories with help from recurrent neural networks. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, page 4311–4312.
- Melissa Roemmele and Andrew S. Gordon. 2015. Creative Help: A Story Writing Assistant. In *Interactive Storytelling*, pages 81–92.
- Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. Automatic keyword extraction from individual documents. *Text Mining: Applications and Theory*, pages 1 – 20.
- Zhihong Shao, Minlie Huang, Jiangtao Wen, Wenfei Xu, and Xiaoyan Zhu. 2019. Long and diverse text generation with planning-based hierarchical variational model. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 3257–3268.
- Anuroop Sriram, Heewoo Jun, Sanjeev Satheesh, and Adam Coates. 2018. Cold fusion: Training seq2seq models together with language models. In *6th International Conference on Learning Representations, ICLR 2018, Workshop Track Proceedings*.
- Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems*, pages 3104–3112.
- Reid Swanson and Andrew S. Gordon. 2012. Say Anything: Using Textual Case-Based Reasoning to Enable Open-Domain Interactive Storytelling. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 2(3):16:1–16:35.
- Pradyumna Tambwekar, Murtaza Dhuliawala, Lara J. Martin, Animesh Mehta, Brent Harrison, and Mark O. Riedl. 2019. Controllable neural story plot generation via reward shaping. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pages 5982–5988.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems*, pages 5998–6008.
- Jingjing Xu, Yi Zhang, Qi Zeng, Xuancheng Ren, Xiaoyan Cai, and Xu Sun. 2018. A skeleton-based model for promoting coherence among sentences in narrative story generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4306–4315.
- Lili Yao, Nanyun Peng, Ralph M. Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. 2019. Plan-and-write: Towards better automatic storytelling. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*, pages 7378–7385.
- Meng-Hsuan Yu, Juntao Li, Danyang Liu, Bo Tang, Haisong Zhang, Dongyan Zhao, and Rui Yan. 2020. Draft and edit: Automatic storytelling through multi-pass hierarchical conditional variational autoencoder. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 1741–1748.

# Liputan6: A Large-scale Indonesian Dataset for Text Summarization

Fajri Koto Jey Han Lau Timothy Baldwin

School of Computing and Information Systems  
The University of Melbourne

fffajri@student.unimelb.edu.au, jeyhan.lau@gmail.com, tbaldwin@unimelb.edu.au

## Abstract

In this paper, we introduce a large-scale Indonesian summarization dataset. We harvest articles from *Liputan6.com*, an online news portal, and obtain 215,827 document-summary pairs. We leverage pre-trained language models to develop benchmark extractive and abstractive summarization methods over the dataset with multilingual and monolingual BERT-based models. We include a thorough error analysis by examining machine-generated summaries that have low ROUGE scores, and expose both issues with ROUGE itself, as well as with extractive and abstractive summarization models.

## 1 Introduction

Despite having the fourth largest speaker population in the world, with 200 million native speakers,<sup>1</sup> Indonesian is under-represented in NLP. One reason is the scarcity of large datasets for different tasks, such as parsing, text classification, and summarization. In this paper, we attempt to bridge this gap by introducing a large-scale Indonesian corpus for text summarization.

Neural models have driven remarkable progress in summarization in recent years, particularly for abstractive summarization. One of the first studies was Rush et al. (2015), where the authors proposed an encoder-decoder model with attention to generate headlines for English Gigaword documents (Graff et al., 2003). Subsequent studies introduced pointer networks (Nallapati et al., 2016b; See et al., 2017), summarization with content selection (Hsu et al., 2018; Gehrmann et al., 2018), graph-based attentional models (Tan et al., 2017), and deep reinforcement learning (Paulus et al., 2018). More recently, we have seen the widespread adoption

of pre-trained neural language models for summarization, e.g. BERT (Liu and Lapata, 2019), BART (Lewis et al., 2020), and PEGASUS (Zhang et al., 2020a).

Progress in summarization research has been driven by the availability of large-scale English datasets, including 320K *CNN/Daily Mail* document-summary pairs (Hermann et al., 2015) and 100k *NYT* articles (Sandhaus, 2008) which have been widely used in abstractive summarization research (See et al., 2017; Gehrmann et al., 2018; Paulus et al., 2018; Lewis et al., 2020; Zhang et al., 2020a). News articles are a natural candidate for summarization datasets, as they tend to be well-structured and are available in large volumes. More recently, English summarization datasets in other flavours/domains have been developed, e.g. *XSum* has 226K documents with highly abstractive summaries (Narayan et al., 2018), *BIGPATENT* is a summarization dataset for the legal domain (Sharma et al., 2019), Reddit TIFU is sourced from social media (Kim et al., 2019), and Cohan et al. (2018) proposed using scientific publications from arXiv and PubMed for abstract summarization.

This paper introduces the first large-scale summarization dataset for Indonesian, sourced from the *Liputan6.com* online news portal over a 10-year period. It covers various topics and events that happened primarily in Indonesia, from October 2000 to October 2010. Below, we present details of the dataset, propose benchmark extractive and abstractive summarization methods that leverage both multilingual and monolingual pre-trained BERT models. We further conduct error analysis to better understand the limitations of current models over the dataset, as part of which we reveal not just modelling issues but also problems with ROUGE.

To summarize, our contributions are: (1) we release a large-scale Indonesian summarization corpus with over 200K documents, an order of mag-

<sup>1</sup><https://www.visualcapitalist.com/100-most-spoken-languages/>.

Example-1	
<p><b>Dokumen:</b> Liputan6.com, Jakarta : Organisasi Negara-negara Pengekspor Minyak (OPEC) mengakui mengalami kesulitan untuk menjaga stabilitas harga minyak dunia. Itu lantaran harga minyak terus melonjak sepanjang tahun ini. Hingga kini harga minyak mentah dunia masih mencapai tingkat tertinggi sejak pecah perang teluk sepuluh tahun silam. [3 kalimat dengan 57 kata tidak ditampilkan] Padahal , sebelumnya OPEC telah merevisi produksi minyak sebanyak tiga kali dalam enam bulan terakhir. Pertama, April hingga Juni dengan kenaikan mencapai 500 ribu barel dan terakhir, September ini, OPEC kembali menaikkan produksi sebesar 800 ribu barel per hari. [5 kalimat dengan 96 kata setelahnya tidak ditampilkan]</p> <p><b>Ringkasan:</b> OPEC kesulitan menjaga stabilitas harga minyak dunia lantaran harga minyak dipasaran terus melonjak. Padahal, OPEC telah tiga kali menaikkan produksi dalam enam bulan terakhir.</p>	<p><b>Document:</b> Liputan6.com, Jakarta: The Organization of Petroleum Exporting Countries (OPEC) has admitted that it is having difficulty maintaining the stability of world oil prices. That's because oil prices continue to soar this year. Until now world crude oil prices have still reached the highest level since the gulf war broke out ten years ago. [3 sentences with 57 words are abbreviated from here] In fact, OPEC had previously revised oil production three times in the last six months. First, April to June with an increase of 500 thousand barrels and last, this September, OPEC has again increased production by 800 thousand barrels per day. [5 sentences with 96 words are abbreviated from here]</p> <p><b>Summary:</b> OPEC is struggling to maintain the stability of world oil prices because oil prices on the market continue to soar. In fact, OPEC has raised production three times in the past six months.</p>
Example-2	
<p><b>Dokumen:</b> Liputan6.com, Jakarta : Gara-gara berusaha kabur saat diminta menunjukkan barang hasil curian, Rosihan bin Usman, tersangka pencurian tas wisatawan asing, baru-baru ini, tersungkur di tembak aparat Kepolisian Resor Denpasar Barat, Bali. Sebelumnya, Rosihan ditangkap massa setelah mencuri tas Nicholas Dreyden, wisatawan asal Inggris. Tas yang berisi dokumen keimigrasian dan surat penting itu diambil Rosihan setelah mengelubui korban. [7 kalimat dengan 78 kata setelahnya tidak ditampilkan]</p> <p><b>Ringkasan:</b> Seorang pencuri tas wisatawan asing ditembak polisi. Ia berusaha kabur saat diminta menunjukkan hasil curian. Karena itu, polisi menembaknya.</p>	<p><b>Document:</b> Liputan6.com, Jakarta: Because of trying to escape when asked to show stolen goods, Rosihan bin Usman, a suspect of the theft of a foreign tourist bag, recently fell down, shot by the West Denpasar Resort Police, Bali. Previously, Rosihan was arrested by the mob after stealing the bag of Nicholas Dreyden, a foreign tourist from England. The bag containing immigration documents and important letters was taken by Rosihan after tricking the victim. [7 sentences with 78 words are abbreviated from here]</p> <p><b>Summary:</b> A foreign tourist bag thief was shot by police. He tried to run away when asked to show the loot. Because of this, the police shot him.</p>

Figure 1: Example articles and summaries from Liputan6. To the left is the original document and summary, and to the right is an English translation (for illustrative purposes). We additionally highlight sentences that the summary is based on (noting that such highlighting is not available in the dataset).

nitude larger than the current largest Indonesian summarization dataset and one of the largest non-English summarization datasets in existence;<sup>2</sup> (2) we present statistics to show that the summaries in the dataset are reasonably abstractive, and provide two test partitions, a standard test set and an extremely abstractive test set; (3) we develop benchmark extractive and abstractive summarization models based on pre-trained BERT models; and (4) we conduct error analysis, on the basis of which we share insights to drive future research on Indonesian text summarization.

## 2 Data Construction

*Liputan6.com* is an online Indonesian news portal which has been running since August 2000, and provides news across a wide range of topics including politics, business, sport, technology, health, and entertainment. According to the Alexa ranking of websites at the time of writing,<sup>3</sup> *Liputan6.com* is ranked 9th in Indonesia and 112th globally. The website produces daily articles along

with a short description for its RSS feed. The summary is encapsulated in the javascript variable `window.kmklabs.article` and the key `shortDescription`, while the article is in the main body of the associated HTML page. We harvest this data over a 10-year window — from October 2000 to October 2010 — to create a large-scale summarization corpus, comprising 215,827 document–summary pairs. In terms of preprocessing, we remove formatting and HTML entities (e.g. &quot;, and \_\_), lowercase all words, and segment sentences based on simple punctuation heuristics. We provide example articles and summaries, with English translations for expository purposes (noting that translations are not part of the dataset), in Figure 1.

As a preliminary analysis of the document–summary pairs over the 10-year period, we binned the pairs into 5 chronologically-ordered groups containing 20% of the data each, and computed the proportion of novel  $n$ -grams (order 1 to 4) in the summary (relative to the source document). Based on the results in Figure 2, we can see that the proportion of novel  $n$ -grams drops over time, implying that the summaries of more recent articles are less

<sup>2</sup>The data can be accessed at [https://github.com/fajri91/sum\\_liputan6](https://github.com/fajri91/sum_liputan6)

<sup>3</sup><https://www.alexa.com/topsites>

Variant	#Doc			% of Novel n-grams			
	Train	Dev	Test	1	2	3	4
Canonical	193,883	10,972	10,972	16.2	52.5	71.8	82.4
Xtreme	193,883	4,948	3,862	22.2	66.7	87.5	96.6

Table 1: Statistics for the canonical and Xtreme variants of our data. The percentage of novel  $n$ -grams is based on the combined Dev and Test set.

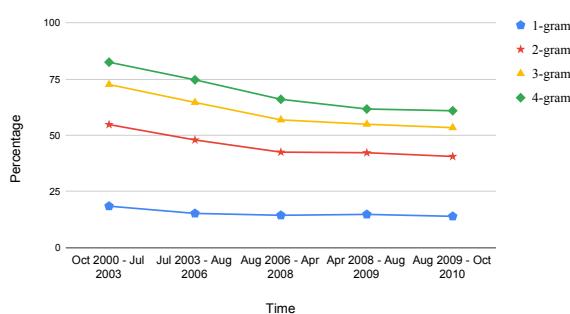


Figure 2: Proportion of novel  $n$ -grams over time in the summaries.

abstractive. For this reason, we decide to use the earlier articles (October 2000 to Jan 2002) as the development and test documents, to create a more challenging dataset. This setup also means there is less topic overlap between training and development/test documents, allowing us to assess whether the summarization models are able to summarize unseen topics.

For the training, development and test partitions, we use a splitting ratio of 90:5:5. In addition to this canonical partitioning of the data, we provide an “Xtreme” variant (inspired by *Xsum*; Narayan et al. (2018)) whereby we discard development and test document–summary pairs where the summary has fewer than 90% novel 4-grams (leaving the training data unchanged), creating a smaller, more challenging data configuration. Summary statistics for the “canonical” and “Xtreme” variants are given in Table 1.

We next present a comparison of Liputan6 (canonical partitioning) and IndoSum (the current largest Indonesian summarization dataset, as detailed in Section 6; Kurniawan and Louvan (2018)) in Table 2. In terms of number of documents, Liputan6 is approximately 11 times larger than IndoSum (the current largest Indonesian summarization dataset), although articles and summaries in Liputan6 are slightly shorter.

To understand the abstractiveness of the summaries in the two datasets, in Table 3 we present

ROUGE scores for the simple baseline of using the first  $N$  sentences as an extractive summary (“LEAD- $N$ ”), and the percentage of novel  $n$ -grams in the summary.<sup>4</sup> We use LEAD-3 and LEAD-2 for IndoSum and Liputan6 respectively, based on the average number of sentences in the summaries (Table 2). We see that Liputan6 has consistently lower ROUGE scores (R1, R2, and RL) for LEAD- $N$ ; it also has a substantially higher proportion of novel  $n$ -grams. This suggests that the summaries in Liputan6 are more abstractive than IndoSum.

To create a ground truth for extractive summarization, we follow Cheng and Lapata (2016) and Nallapati et al. (2016a) in greedily selecting the subset of sentences in the article that maximizes the ROUGE score based on the reference summary. As a result, each sentence in the article has a binary label to indicate whether they should be included as part of an extractive summary. Extractive summaries created this way will be referred to as “ORACLE”, to denote the upper bound performance of an extractive summarization system.

### 3 Summarization Models

We follow Liu and Lapata (2019) in building extractive and abstractive summarization models using BERT as an encoder to produce contextual representations for the word tokens. The architecture of both models is presented in Figure 3. We tokenize words with WordPiece, and append [CLS] (prefix) and [SEP] (suffix) tokens to each sentence. To further distinguish the sentences, we add even/odd segment embeddings ( $T_A/T_B$ ) based on the order of the sentence to the word embeddings. For instance, for a document with sentences  $[s_1, s_2, s_3, s_4]$ , the segment embeddings are  $[T_A, T_B, T_A, T_B]$ . Position embeddings ( $P$ ) are also used to denote the position of each token. The WordPiece, segment, and position embeddings are summed together and provided as input to BERT.

BERT produces a series of contextual representations for the word tokens, which we feed into a (second) transformer encoder/decoder for the extractive/abstractive summarization model. We detail the architecture of these two models in Sections 3.1 and 3.2. Note that this second transformer is initialized with random parameters (i.e. it is not pre-trained).

For the pre-trained BERT encoder, we use mul-

<sup>4</sup>All statistics are based on the entire dataset, encompassing the training, dev, and test data.

Dataset	#Doc			Article			Summary		
	Train	Dev	Test	$\mu(\text{Word})$	$\mu(\text{Sent})$	#Vocab	$\mu(\text{Word})$	$\mu(\text{Sent})$	#Vocab
IndoSum	14,252	750	3,762	347.23	18.37	117K	68.09	3.47	53K
Liputan6	193,883	10,972	10,972	232.91	12.60	311K	30.43	2.09	100K

Table 2: A comparison of IndoSum and Liputan6.  $\mu(\text{Word})$  and  $\mu(\text{Sent})$  denote the average number of words and sentences, respectively.

Dataset	LEAD-N			% of Novel n-grams			
	R1	R2	RL	1	2	3	4
IndoSum	65.6	58.9	64.8	3.1	10.8	16.2	20.3
Liputan6	41.2	27.1	38.7	12.9	41.6	57.6	66.9

Table 3: Abstractiveness of the summaries in IndoSum and Liputan6.

tilingual BERT (mBERT) and our own IndoBERT (Koto et al., to appear).<sup>5</sup> IndoBERT is a BERT-Base model we trained ourselves using Indonesian documents from three sources: (1) Indonesian Wikipedia (74M words); (2) news articles (55M words) from Kompas,<sup>6</sup> Tempo (Tala et al., 2003),<sup>7</sup> and Liputan6,<sup>8</sup> and (3) the Indonesian Web Corpus (90M words; Medved and Suchomel (2017)). In total, the training data has 220M words. We implement IndoBERT using the Huggingface framework,<sup>9</sup> and follow the default configuration of BERT-Base (uncased): hidden size = 768d, hidden layers = 12, attention heads = 12, and feed-forward = 3,072d. We train IndoBERT with 31,923 Word-Pieces (vocabulary) for 2 million steps.

### 3.1 Extractive Model

After the document is processed by BERT, we have a contextualized embedding for every word token in the document. To learn inter-sentential relationships, we use the [CLS] embeddings ( $[x_{S_1}, x_{S_2}, \dots, x_{S_m}]$ ) to represent the sentences, to which we add a sentence-level positional embedding ( $P$ ), and feed them to a transformer encoder (Figure 3). An MLP layer with sigmoid activation is applied to the output of the transformer encoder to predict whether a sentence should be extracted (i.e.  $\tilde{y}_S \in \{0, 1\}$ ). We train the model with binary

<sup>5</sup>The pre-trained mBERT is sourced from: <https://github.com/google-research/bert>.

<sup>6</sup><https://kompas.com>

<sup>7</sup><https://koran.tempo.co>

<sup>8</sup>For Liputan6, we use only the articles from the training partition.

<sup>9</sup><https://huggingface.co/>

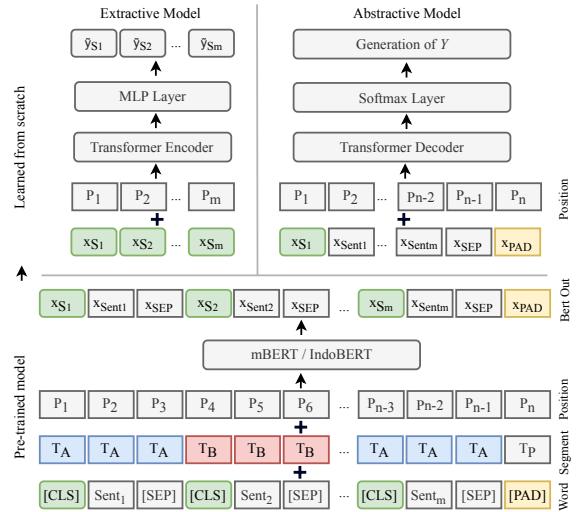


Figure 3: Architecture of the extractive and abstractive summarization models.

cross entropy, and update all model parameters (including BERT) during training. Note that the parameters in the transformer encoder and the MLP layer are initialized randomly, and learned from scratch.

The transformer encoder is configured as follows: layers = 2, hidden size = 768, feed-forward = 2,048, and heads = 8. In terms of training hyperparameters, we train using the Adam optimizer with learning rate  $lr = 2e^{-3} \cdot \min(\text{step}^{-0.5}, \text{step} \cdot \text{warmup}^{-1.5})$  where warmup = 10,000. We train for 50,000 steps on  $3 \times \text{V100}$  16GB GPUs, and perform evaluation on the development set every 2,500 steps. At test time, we select sentences for the extractive summary according to two conditions: the summary must consist of: (a) at least two sentences, and (b) at least 15 words. These values were set based on the average number of sentences and the minimum number of words in a summary. We also apply trigram blocking to reduce redundancy (Paulus et al., 2018). Henceforth, we refer to this model as ‘‘BERTTEXT’’.

### 3.2 Abstractive Model

Similar to the extractive model, we have a second transformer to process the contextualized embeddings from BERT. In this case, we use a transformer decoder instead (i.e. an attention mask is used to prevent the decoder from attending to future time steps), as we are learning to generate an abstractive summary. But unlike the extractive model, we use the BERT embeddings for all tokens as input to the transformer decoder (as we do not need sentence representations). We add to these BERT embeddings a second positional encoding before feeding them to the transformer decoder (Figure 3). The transformer decoder is initialized with random parameters (i.e. no pre-training).

The transformer decoder is configured as follows: layers = 6, hidden size = 768, feed-forward = 2,048, and heads = 8. Following Liu and Lapata (2019), we use a different learning rate for BERT and the decoder when training the model:  $lr = 2e^{-3} \cdot \min(\text{step}^{-0.5}, \text{step} \cdot 20,000^{-1.5})$  and  $0.1 \cdot \min(\text{step}^{-0.5}, \text{step} \cdot 10,000^{-1.5})$  for BERT and the transformer decoder, respectively. Both networks are trained with the Adam optimizer for 200,000 steps on 4×V100 16GB GPUs and evaluated every 10,000 steps. For summary generation, we use beam width = 5, trigram blocking, and a length penalty (Wu et al., 2016) to generate at least two sentences and at least 15 words (similar to the extractive model).

Henceforth the abstractive model will be referred to as “BERTABS”. We additionally experiment with a third variant, “BERTTEXTABS”, where we use the weights of the fine-tuned BERT in BERTTEXT for the encoder (instead of off-the-shelf BERT weights).

## 4 Experiment and Results

We use three ROUGE (Lin, 2004) F-1 scores as evaluation metrics: R1 (unigram overlap), R2 (bigram overlap), and RL (longest common subsequence overlap). In addition, we also provide BERTSCORE (F-1), as has recently been used for machine translation evaluation (Zhang et al., 2020b).<sup>10</sup> We use the development set to select the best checkpoint during training, and report the evaluation scores for the canonical and Xtreme test sets in Table 4. For both test sets, the summarization models are trained using the same training

set, but they are tuned with a different development set (see Section 2 for details). In addition to the BERT models, we also include two pointer-generator models (See et al., 2017): (1) the base model (PTGEN); and (2) the model with coverage penalty (PTGEN+COV).<sup>11</sup>

We first look at the baseline LEAD- $N$  and ORACLE results. LEAD-2 is the best LEAD- $N$  baseline for Liputan6. This is unsurprising, given that in Table 2, the average summary length was 2 sentences. We also notice there is a substantial gap between ORACLE and LEAD-2: 12–15 points for R1 and 5–7 points for BERTSCORE, depending on the test set. This suggests that the baseline of using the first few sentences as an extractive summary is ineffective. Comparing the performance between the canonical and Xtreme test sets, we see a substantial drop in performance for both LEAD- $N$  and ORACLE, highlighting the difficulty of the Xtreme test set due to its increased abstractiveness.

For the pointer-generator models, we see little improvement when including the coverage mechanism (PTGEN+COV vs. PTGEN), implying that there is minimal repetition in the output of PTGEN. We suspect this is due to the Liputan6 summaries being relatively short (2 sentences with 30 words on average). A similar observation is reported by Narayan et al. (2018) for XSum, where the summaries are similarly short (a single sentence with 23 words, on average).

Next we look at the BERT models. Overall they perform very well, with both the mBERT and IndoBERT models outperforming the LEAD- $N$  baselines and PTGEN models by a comfortable margin. IndoBERT is better than mBERT (approximately 1 ROUGE point better on average over most metrics), showing that a monolingually-trained BERT is a more effective pre-trained model than the multilingual variant. The best performance is achieved by IndoBERT’s BERTTEXTABS. In the canonical test set, the improvement over LEAD-2 is +4.4 R1, +2.62 R2, +4.3 R3, and +3.4 BERTSCORE points. In the Xtreme test set, BERTTEXTABS suffers a substantial drop compared to the canonical test set (6–7 ROUGE and 2 BERTSCORE points), although the performance gap between it and LEAD-2 is about the same.

<sup>10</sup>[https://github.com/Tiiiger/bert\\_score](https://github.com/Tiiiger/bert_score)

<sup>11</sup>We use the default hyper-parameter configuration recommended by the original authors for the pointer-generator models.

Model	Canonical Test Set				Xtreme Test Set			
	R1	R2	RL	BS	R1	R2	RL	BS
LEAD-1	32.67	18.50	29.40	72.62	27.27	11.56	23.60	71.19
LEAD-2	36.68	20.23	33.71	74.58	31.10	12.78	27.63	72.98
LEAD-3	34.49	18.84	32.06	74.31	29.54	12.05	26.68	72.78
ORACLE	51.54	30.56	47.75	79.24	43.69	18.57	38.84	76.75
PTGEN	36.10	19.19	33.56	75.92	30.41	12.05	27.51	74.10
PTGEN+Cov	35.53	18.56	32.92	75.75	30.27	11.81	27.26	74.11
BERTTEXT (mBERT)	37.51	20.15	34.57	75.22	31.83	12.63	28.37	73.62
BERTABS (mBERT)	39.48	21.59	36.72	77.19	33.26	13.82	30.12	75.40
BERTTEXTABS (mBERT)	39.81	21.84	37.02	77.39	33.86	14.13	30.73	75.69
BERTTEXT (IndoBERT)	38.03	20.72	35.07	75.33	31.95	12.74	28.47	73.64
BERTABS (IndoBERT)	40.94	<b>23.01</b>	37.89	77.90	34.59	<b>15.10</b>	31.19	75.84
BERTTEXTABS (IndoBERT)	<b>41.08</b>	22.85	<b>38.01</b>	<b>77.93</b>	<b>34.84</b>	15.03	<b>31.40</b>	<b>75.99</b>

Table 4: ROUGE results for the canonical and Xtreme test sets. All ROUGE (“R1”, “R2”, and “RL”) scores have a confidence interval of at most  $\pm 0.3$ , as reported by the official ROUGE script. “BS” is BERSCORE computed with `bert-base-multilingual-cased` (layer 9), as suggested by Zhang et al. (2020b).

## 5 Error Analysis

In this section, we analyze errors made by the extractive (BERTTEXT) and abstractive (BERTTEXTABS) models to better understand their behaviour. We use the mBERT version of these models in our analysis.<sup>12</sup>

### 5.1 Error Analysis of Extractive Summaries

We hypothesized that the disparity between ORACLE and BERTTEXT (14.03 point difference for R1 in the canonical test set) was due to the number of extracted sentences. To test this, when extracting sentences with BERTTEXT, we set the total number of extracted sentences to be the same as the number of sentences in the ORACLE summary. However, we found minimal benefit using this approach, suggesting that the disparity is not a result of the number of extracted sentences.

To investigate this further, we present the frequency of *sentence positions* that are used in the summary in ORACLE and BERTTEXT for the canonical test set in Figure 4a. We can see that BERTTEXT tends to over-select the first two sentences as the summary. In terms of proportion, 65.47% of

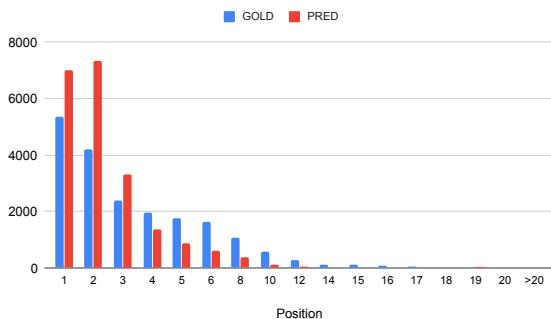
BERTTEXT summaries involve the first two sentences. In comparison, only 42.54% of ORACLE summaries use sentences in these positions. One may argue that this is because the training and test data have different distributions under our chronological partitioning strategy (recall that the test set is sampled from the earliest articles), but that does not appear to be the case: as Figure 4b shows, the distribution of sentence positions in the training data is very similar to the test data — 43.14% of ORACLE summaries involve the first two sentences.

### 5.2 Error Analysis of Abstractive Summaries

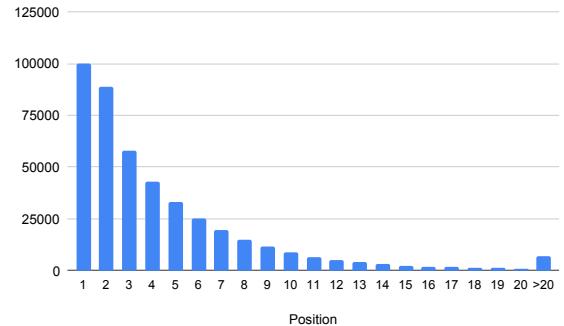
To perform error analysis for BERTTEXTABS, we randomly sample 100 documents with an R1 score  $< 0.4$  in the canonical test set (which accounts for nearly 50% of the test documents). Two native Indonesian speakers examined these 100 samples to manually assess the quality of the summaries, and score them on a 3-point ordinal scale: (1) *bad*; (2) *average*; and (3) *good*. Each annotator is presented with the source document, the reference summary, and the summary generated by BERTTEXTABS. In addition to the overall quality evaluation, we also asked the annotators to analyze a number of (fine-grained) attributes in the summaries:

- *Abbreviations*: the system summary uses abbreviations that are different to the reference summary.

<sup>12</sup>The error analysis is based on mBERT rather than IndoBERT simply because this was the best-performing model at the time the error analysis was performed. While IndoBERT ultimately performed slightly better, given that the two models are structurally identical, we would expect to see a similar pattern of results.



(a) Distribution of sentence positions for ORACLE and BERTTEXT in the canonical test set.



(b) Distribution of sentence positions for ORACLE in the training set.

Figure 4: Position of ORACLE and/or Predicted Extractive Summaries

Category	Bad	Avg.	Good
#Samples (100)	32	8	60
Abbreviation (%)	21.9	25.0	40.0
Morphology (%)	12.5	25.0	36.7
Paraphrasing (%)	50.0	87.5	<b>86.7</b>
Lack of coverage (%)	<b>90.6</b>	<b>100.0</b>	40.0
Wrong focus (%)	68.8	0.00	8.3
Un. details (from doc) (%)	<b>90.6</b>	75.0	75.0
Un. details (not from doc) (%)	18.8	12.5	5.0

Table 5: Error analysis for 100 samples with  $R1 < 0.4$ .

- *Morphology*: the system summary uses morphological variants of the same lemmas contained in the reference summary.
- *Synonyms/paraphrasing*: the system summary contains paraphrases of the reference summary.
- *Lack of coverage*: the system summary lacks coverage of certain details that are present in the reference summary.
- *Wrong focus*: the system summarizes a different aspect/focus of the document to the reference summary.
- *Unnecessary details (from document)*: the system summary includes unimportant but factually correct information.
- *Unnecessary details (not from document)*: the system summary includes unimportant and factually incorrect information (hallucinations).

We present a breakdown of the different error types in Table 5. Inter-annotator agreement for the overall quality assessment is high (Pearson’s  $r = 0.69$ ). Disagreements in the quality label (*bad*,

*average, good*) are resolved as follows: (1)  $\{bad, average\} \rightarrow bad$ ; and (2)  $\{good, average\} \rightarrow good$ . We only have four examples with  $\{bad, good\}$  disagreement, which we resolved through discussion. Interestingly, more than half (60) of our samples were found to have *good* summaries. The primary reasons why these summaries have low ROUGE scores are paraphrasing (86.7%), and the inclusion of additional (but valid) details (75.0%). Abbreviations and morphological differences also appear to be important factors. These results underline a problem with the ROUGE metric, in that it is unable to detect good summaries that use a different set of words to the reference summary. One way forward is to explore metrics that consider sentence semantics beyond word overlap such as METEOR (Banerjee and Lavie, 2005) and BERTSCORE,<sup>13</sup> and question-answering system based evaluation such as APES (Eyal et al., 2019) and QAGS (Wang et al., 2020). Another way is to create more reference summaries (which will help with the issue of the system summaries including [validly] different details to the single reference).

Looking at the results for *average* summaries (middle column), BERTTEXTABS occasionally fails to capture salient information: 100% of the summaries have coverage issues, and 75.0% contain unnecessary (but valid) details. They also tend to use paraphrases (87.5%), which further impacts on a lower ROUGE score. Finally, the *bad* system summaries have similar coverage issues, and also tend to have a very different focus compared to the

<sup>13</sup>Indeed, we suggest that BERTSCORE should be used as the canonical evaluation metric for the dataset, but leave empirical validation of its superiority for Indonesian summarization evaluation to future work.

Example-1 of error analysis (Abbreviation, morphoplogy, synonyms/paraphrasing, and details from the document)	
<b>Dokumen:</b> Liputan6.com , Jakarta : Protes masih bergema menyambut Keputusan Menteri Tenaga Kerja dan Transmigrasi Nomor 78 Tahun 2001 . Kebijakan yang sengaja dikeluarkan sebagai wujud perubahan keputusan sebelumnya ini , sampai sekarang , masih mengundang kecaman keras dari pekerja di Indonesia . Itulah sebabnya , mereka menuntut Kepmenakertrans baru ini dicabut karena dinilai merugikan pekerja . [19 kalimat dengan 406 kata tidak ditampilkan] Sementara itu , SPSI secara tegas menolak segala bentuk negosiasi . [3 kalimat dengan 45 kata setelahnya tidak ditampilkan]	<b>Document:</b> Liputan6.com, Jakarta: Protests still resonate with welcoming Minister of Manpower and Transmigration Decree No. 78/2001. This policy, which was deliberately issued as an amendment to the previous decision, until now, still invites harsh criticism from workers in Indonesia. That is why they demand to revoke the new Kepmenakertrans because it is considered detrimental to workers. [19 sentences with 406 words are abbreviated from here] Meanwhile, SPSI firmly rejected all forms of negotiation. [3 sentences with 45 words are abbreviated from here]
<b>Ringkasan manusia:</b> pemberlakuan <b>kepmenakertrans</b> 78/2001 masih mengundang rasa tidak puas di dada sejumlah pekerja indonesia . maka , lahirlah <b>tuntutan</b> agar peraturan yang dinilai merugikan ini dicabut .	<b>Gold Summary:</b> The enactment of <b>Kepmenakertrans</b> 78/2001 still invites the dissatisfaction of Indonesian workers. hence, demands to revoke the regulation arose as it was considered to be detrimental.
<b>Ringkasan sistem [Good]:</b> keputusan menteri tenaga kerja dan transmigrasi nomor 78 tahun 2001 mengundang kecaman keras dari pekerja di indonesia . mereka menuntut kepmenakertrans dicabut karena dinilai merugikan pekerja . spsi menolak negosiasi .	<b>System Summary [Good]:</b> Minister of Manpower and Transmigration Decree number 78 of 2001 invited strong criticism from workers in Indonesia. They demand to revoke Kepmenakertrans because it is considered detrimental to workers. SPSI rejects negotiations.
Example-2 of error analysis (Lack of coverage, wrong focus, and details that are not from the document)	
<b>Dokumen:</b> Liputan6.com , Jakarta : Langkah reshuffle yang dilakukan Presiden Abdurrahman Wahid , agaknya tak mendapat restu . Buktiunya , Wakil Presiden Megawati Sukarnoputri kembali tidak hadir dalam pelantikan tiga menteri bidang ekonomi , Rabu ( 13 / 6 ) . [8 kalimat dengan 113 kata setelahnya tidak ditampilkan]	<b>Document:</b> Liputan6.com, Jakarta: The reshuffle step was taken by President Abdurrahman Wahid, apparently did not get the blessing. The proof, Vice President Megawati Sukarnoputri was again not present at the inauguration of three ministers in the economic sector, Wednesday (6/13). [8 sentences with 113 words are abbreviated from here]
<b>Ringkasan manusia:</b> wapres megawati sukarnoputri , kembali tidak hadir dalam pelantikan tiga menteri baru . dalam reshuffle 1 juni , megawati juga tak muncul dalam pelantikan , karena merasa tak dilibatkan dalam reshuffle kabinet .	<b>Gold Summary:</b> Vice President Megawati Sukarnoputri, is not present at the inauguration of three new ministers again. In the reshuffle on June 1, Megawati also did not appear in the inauguration, because she felt not involved in the cabinet reshuffle.
<b>Ringkasan sistem [Bad]:</b> presiden abdurrahman wahid kembali tidak hadir dalam pelantikan tiga menteri bidang ekonomi . ketidaksepakatan soal perombakan kabinet itu juga terjadi 1 juni silam . presiden meminta mereka lebih menjaga koordinasi antarmenteri .	<b>System Summary [Bad]:</b> President Abdurrahman Wahid was again absent from the inauguration of three ministers in the economic sector. disagreement about the cabinet reshuffle also occurred 1 June ago. the president asked them to maintain more coordination between ministries.

Figure 5: Two examples to highlight error categories used in our error analysis.

reference summary (90.6%).

In Figure 5 we show two representative examples from BERTTEXTABS. The first example is considered *good* by our annotators, but due to abbreviations, morphological differences, paraphrasing, and additional details compared to the reference summary, the ROUGE score is <0.4. In this example, the gold summary uses the abbreviation *kepmenakertrans* while BERTTEXTABS generates the full phrase *keputusan menteri tenaga kerja dan transmigrasi* (which is correct). The example also uses paraphrases (*invites strong criticism* to explain *dissatisfaction*), and there are morphological differences in words such as *tuntutan* (noun) vs. *menuntut* (verb). The low ROUGE score here highlights the fact that the bigger issue is with ROUGE itself rather than the summary.

The second example is considered to be *bad*, with the following issues: lack of coverage, wrong focus, and contains unnecessary details that are not from the article. The first sentence *President Abdurrahman Wahid was absent* has nothing to do

with the original article, creating a different focus (and confusion) in the overall summary.

To summarize, coverage, focus, and the inclusion of other details are the main causes of low quality summaries. Our analysis reveals that abbreviations and paraphrases are another cause of summaries with low ROUGE scores, but that is an issue with ROUGE rather than the summaries. Encouragingly, hallucination (generating details not in the original document) is not a major issue for these models (notwithstanding that almost 20% of *bad* samples contain hallucinations).

## 6 Related Datasets

Previous studies on Indonesian text summarization have largely been extractive and used small-scale datasets. [Gunawan et al. \(2017\)](#) developed an unsupervised summarization model over 3K news articles using heuristics such as sentence length, keyword frequency, and title features. In a similar vein, [Najibullah \(2015\)](#) trained a naive Bayes model to extract summary sentences in a 100-article dataset.

Aristoteles et al. (2012) and Silvia et al. (2014) apply genetic algorithms to a summarization dataset with less than 200 articles. These studies do not use ROUGE for evaluation, and the datasets are not publicly available.

Koto (2016) released a dataset for chat summarization by manually annotating chat logs from WhatsApp.<sup>14</sup> However, this dataset contains only 300 documents. The largest summarization data to date is *IndoSum* (Kurniawan and Louvan, 2018), which has approximately 19K news articles with manually-written summaries. Based on our analysis, however, the summaries of *IndoSum* are highly extractive.

Beyond Indonesian, there is only a handful of non-English summarization datasets that are of sufficient size to train modern deep learning summarization methods over, including: (1) LCSTS (Hu et al., 2015), which contains 2 million Chinese short texts constructed from the Sina Weibo microblogging website; and (2) ES-News (Gonzalez et al., 2019), which comprises 270k Spanish news articles with summaries. LCSTS documents are relatively short (less than 140 Chinese characters), while ES-News is not publicly available. Our goal is to create a benchmark corpus for Indonesian text summarization that is both large scale and publicly available.

## 7 Conclusion

We release Liputan6, a large-scale summarization corpus for Indonesian. Our dataset comes with two test sets: a canonical test set and an “Xtreme” variant that is more abstractive. We present results for several benchmark summarization models, in part based on IndoBERT, a new pre-trained BERT model for Indonesian. We further conducted extensive error analysis, as part of which we identified a number of issues with ROUGE-based evaluation for Indonesian.

## Acknowledgments

We are grateful to the anonymous reviewers for their helpful feedback and suggestions. In this research, Fajri Koto is supported by the Australia Awards Scholarship (AAS), funded by the Department of Foreign Affairs and Trade (DFAT), Australia. This research was undertaken using the LIEF HPC-GPGPU Facility hosted at The University of

Melbourne. This facility was established with the assistance of LIEF Grant LE170100200.

## References

- Aristoteles Aristoteles, Yeni Herdiyeni, Ahmad Ridha, and Julio Adisantoso. 2012. Text feature weighting for summarization of document Bahasa Indonesia using genetic algorithm. *IJCSI International Journal of Computer Science Issues*, 9(1):1–6.
- Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72.
- Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 484–494.
- Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *NAACL HLT 2018: 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, volume 2, pages 615–621.
- Matan Eyal, Tal Baumel, and Michael Elhadad. 2019. Question answering as an automatic evaluation metric for news article summarization. In *NAACL-HLT 2019: Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 3938–3948.
- Sebastian Gehrmann, Yuntian Deng, and Alexander M Rush. 2018. Bottom-up abstractive summarization. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 4098–4109.
- J.-A. Gonzalez, L.-F. Hurtado, E. Segarra, F. Garcia-Granada, and E. Sanchis. 2019. Summarization of Spanish talk shows with siamese hierarchical attention networks. *Applied Sciences*, 9(18).
- David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2003. English Gigaword. Linguistic Data Consortium.
- D Gunawan, A Pasaribu, R F Rahmat, and R Budiarto. 2017. Automatic text summarization for Indonesian language using TextTeaser. *IOP Conference Series: Materials Science and Engineering*, 190(1):12048.
- Karl Moritz Hermann, Tomas Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. *Neural Information Processing Systems*, pages 1693–1701.

<sup>14</sup><https://www.whatsapp.com/>.

- Wan Ting Hsu, Chieh-Kai Lin, Ming-Ying Lee, Kerui Min, Jing Tang, and Min Sun. 2018. A unified model for extractive and abstractive summarization using inconsistency loss. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 132–141.
- Baotian Hu, Qingcai Chen, and Fangze Zhu. 2015. LC-STS: A large scale Chinese short text summarization dataset. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1967–1972.
- Byeongchang Kim, Hyunwoo Kim, and Gunhee Kim. 2019. Abstractive summarization of Reddit posts with multi-level memory networks. In *NAACL-HLT 2019: Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 2519–2531.
- Fajri Koto. 2016. A publicly available Indonesian corpora for automatic abstractive and extractive chat summarization. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016)*.
- Fajri Koto, Afshin Rahimi, Jey Han Lau, and Timothy Baldwin. to appear. IndoLEM and IndoBERT: A benchmark dataset and pre-trained language model for Indonesian NLP. In *Proceedings of the 28th International Conference on Computational Linguistics (COLING 2020)*.
- Kemal Kurniawan and Samuel Louvan. 2018. Indo-sum: A new benchmark dataset for Indonesian text summarization. In *2018 International Conference on Asian Language Processing (IALP)*, pages 215–220.
- Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.
- Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81.
- Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *2019 Conference on Empirical Methods in Natural Language Processing*, pages 3728–3738.
- Marek Medved and Vít Suchomel. 2017. Indonesian web corpus (idWac). In *LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (FAL), Faculty of Mathematics and Physics, Charles University*.
- Ahmad Najibullah. 2015. Indonesian text summarization based on naive Bayes method. *Proceeding Of The International Seminar and Conference 2015*, 1(1).
- Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2016a. SummaRuNNer: A recurrent neural network based sequence model for extractive summarization of documents. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)*, pages 3075–3081.
- Ramesh Nallapati, Bowen Zhou, Cicero Nogueira dos santos, Caglar Gulcehre, and Bing Xiang. 2016b. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290.
- Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don’t give me the details, just the summary! Topic-aware convolutional neural networks for extreme summarization. In *EMNLP 2018: 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807.
- Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. In *Proceedings of the 6th International Conference on Learning Representations*.
- Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 379–389.
- Evan Sandhaus. 2008. The New York Times annotated corpus. Linguistic Data Consortium.
- Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 1073–1083.
- Eva Sharma, Chen Li, and Lu Wang. 2019. BIG-PATENT: A large-scale dataset for abstractive and coherent summarization. In *ACL 2019: The 57th Annual Meeting of the Association for Computational Linguistics*, pages 2204–2213.
- Silvia, Pitri Rukmana, Vivi Regina Aprilia, Derwin Suhartono, Rini Wongso, and Meiliana. 2014. Summarizing text for Indonesian language by using latent Dirichlet allocation and genetic algorithm. In *1st International Conference on Electrical Engineering, Computer Science and Informatics 2014*, pages 148–153.
- F. Tala, J. Kamps, K.E. Mller, and M. de Rijke. 2003. The impact of stemming on information retrieval in Bahasa Indonesia. In *The 14th Meeting of Computational Linguistics in the Netherlands*.

Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. 2017. Abstractive document summarization with a graph-based attentional neural model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1171–1181.

Alex Wang, Kyunghyun Cho, and Mike Lewis. 2020. Asking and answering questions to evaluate the factual consistency of summaries. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5008–5020.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, ukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google’s neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020a. PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. In *ICML 2020: 37th International Conference on Machine Learning*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020b. BERTScore: Evaluating text generation with BERT. In *ICLR 2020: Eighth International Conference on Learning Representations*.

# Generating Sports News from Live Commentary: A Chinese Dataset for Sports Game Summarization

Kuan-Hao Huang<sup>1</sup> Chen Li<sup>2</sup> Kai-Wei Chang<sup>1</sup>

<sup>1</sup>University of California, Los Angeles

<sup>2</sup>Tencent AI Lab

<sup>1</sup>{khhuang, kwchang}@cs.ucla.edu

<sup>2</sup>lichen.lc18@gmail.com

## Abstract

Sports game summarization focuses on generating news articles from live commentaries. Unlike traditional summarization tasks, the source documents and the target summaries for sports game summarization tasks are written in quite different writing styles. In addition, live commentaries usually contain many named entities, which makes summarizing sports games precisely very challenging. To deeply study this task, we present SPORTSSUM<sup>1</sup>, a Chinese sports game summarization dataset which contains 5,428 soccer games of live commentaries and the corresponding news articles. Additionally, we propose a two-step summarization model consisting of a *selector* and a *rewriter* for SPORTSSUM. To evaluate the correctness of generated sports summaries, we design two novel score metrics: *name matching score* and *event matching score*. Experimental results show that our model performs better than other summarization baselines on ROUGE scores as well as the two designed scores.

## 1 Introduction

There are a large number of sports games playing every day. Apparently, manually writing sports news articles to summarize every game is labor-intensive and infeasible. How to automatically generate sports summaries, therefore, becomes a popular and demanding task. Recently, generating news from live commentaries has gradually attracted attention in the academic community (Zhang et al., 2016; Yao et al., 2017). At the same time, several trials have been done in the industry such as sports news from Toutiao’s Xiaoming Bot<sup>2</sup>, Sohu Ruibao<sup>3</sup> and AI football news<sup>4</sup>.

<sup>1</sup>The dataset is available at <https://github.com/ej0c16/Sportssum>

<sup>2</sup><http://www.nbd.com.cn/columns/803>

<sup>3</sup><https://mp.sohu.com/profile?xpt=c29odW1wMzZpdDlzQHNvaHUuY29t>

<sup>4</sup><https://www.51zhanbao.com>

Live Commentary		
Time	Scores	Commentary Sentence
66'	0-0	多特蒙德球员格策拼抢犯规,对手获得控球权。Dortmund's player Götze fouled, and the opponent got the possession of the ball.
66'	0-0	施魏因斯泰格为拜仁慕尼黑赢得一个任意球。Schweinsteiger got a free kick for Bayern Munich.
67'	1-0	进球啦！！！拜仁慕尼黑球员克罗斯大禁区外左脚射门,球从右下角飞进球门,球进了!助攻的是穆勒。拜仁慕尼黑1-0 多特蒙德。Goal!!! Bayern Munich's player Kroos shot with his left foot from the outside of the penalty area. The ball flew into the goal through the lower right corner. The ball went in! Muller gave the assist. Bayern Munich 1-0 Dortmund.
71'	1-0	拜仁慕尼黑球员里贝里大禁区左侧尝试右脚射门,可惜皮球高出球门,给他传球的是拉姆。Bayern Munich's player Ribery tried to shoot with his right foot from the penalty area's left side, but the ball was higher than the crossbar. Lahm passed the ball to him.

Sports News Article		
开场3分钟，克罗斯左侧任意球被顶到后点，里贝里禁区边缘抽射偏出近门柱。第8分钟，穆勒右路与曼朱基奇打出踢墙配合，在门前12米处推射被苏博蒂奇铲出底线。第13分钟，里贝里右路塞球，克罗斯在门前27米处抽射偏出近门柱。(...)	In the 3rd minutes, Kroos's free kick on the left was tipped to the back, and Ribery's shot from the penalty area missed. In the 8th minute, Muller and Mandžukić had teamwork, and Müller's shot from the 12 meters ahead the goal line was touched out by Subotić. In the 13th minute, Ribery passed the ball from the right, and Kroos's shot near the 27 meters ahead the goal line missed. (...)	

Table 1: An example of SPORTSSUM dataset.

Unlike traditional text summarization tasks (Hermann et al., 2015; Rush et al., 2015), the source documents and the target summaries for sports game summarization tasks are written in quite different styles. Live commentaries are the real-time transcripts of the commentators. Accordingly, commentary sentences are more colloquial and informal. In contrast, news summaries are usually more narrative and well-organized since they are written after the games. In addition, commentaries contain a large number of player names. One player can be referred to multiple times in the whole game, and one commentary sentence may mention multiple player names simultaneously. Those properties

make sports games summarization tasks very challenging.

In this paper, we present **SPORTSSUM**, a Chinese dataset for studying sports game summarization tasks. We collect 5,428 pairs of live commentaries and news articles from seven famous soccer leagues. To the best of our knowledge, **SPORTSSUM** is the largest Chinese sports game summarization dataset. In addition, we propose a two-step summarization model for **SPORTSSUM**, which learns a *selector* to extract important commentary sentences and trains a *rewriter* to convert the selected sentences to a news article. To encourage the model to capture the relations between players and actions better, we replace all the player names in the training sentences with a special token and train the proposed model on the modified template-like sentences.

The proposed model performs better than existing extractive and abstractive summarization baseline models in ROUGE scores (Lin, 2004). However, we observe that ROUGE scores cannot evaluate the correctness of generated summaries very well. Therefore, we design two new scores, *name matching score* and *event matching score*, as the auxiliary metrics for **SPORTSSUM**. Our experimental results demonstrate that the proposed model is superior to the baseline models in all the metrics.

Summarizing documents between two articles written in different styles and involving many named entities is not limited to the sports game summarization tasks. There are many possible applications, such as summarizing events from tweets and summarizing trends from forum comments. We hope that **SPORTSSUM** provides a potential research platform to develop advanced techniques for this type of summarization tasks.

## 2 Dataset

We present **SPORTSSUM**, a sports game summarization dataset in Chinese.

**Data collection.** We crawl the records of soccer games from Sina Sports Live<sup>5</sup>. The collected records contain soccer games in seven different leagues (Bundesliga, CSL, Europa, La Liga, Ligue 1, PL, Series A, UCL) from 2012 to 2018. For each game, we have a live commentary document  $C$  and a news article  $R$ , as illustrated in Table 1. The live commentary document  $C$  con-

League	# of games
Bundesliga	453
CSL	1371
Europa	143
La Liga	713
Ligue 1	161
Premier League	1220
Serie A	890
UCL	477
All	5428

Table 2: The number of games in different leagues.

Source	Avg. # chars	Avg. # words	Avg. # sent.	Total # vocab
Commentary	3459.97	1825.63	193.77	43482
News	801.11	427.98	23.80	21294

Table 3: Statistics of **SPORTSSUM** dataset.

sists of a series of tuples  $(t_i, s_i, c_i)$ , where  $t_i$  is the timeline information,  $s_i$  represents the current scores, and  $c_i$  denotes the commentary sentence. The news article  $R$  consists of several news sentences  $r_i$ . In addition to commentaries and news reports, we also include some metadata, such as rosters, starting lineups, and player positions, which is potentially helpful for sports game summarization tasks.

**Data cleaning.** The crawled live commentary documents and news articles are quite noisy. Therefore, we apply multiple steps of data cleaning to improve the quality of the dataset. We first remove all the HTML tags from the commentary documents and the news articles. Then, we observe that there are usually some descriptions that cannot be directly inferred from the commentaries at the beginning of news articles, such as matching history. Hence, we design a heuristic rule to remove those descriptions. We identify several *starting keywords* which can indicate the start of a game, such as “—开场(at the beginning of the game)” and “开场后(after the game started)”. The full list of starting keywords can be found in Appendix A. Once we see a starting keyword appearing in a news report, we remove all the sentences before the starting keyword. Finally, we discard those games with the number of news sentences being less than 5 and the number of commentary sentences being less than 20. After data cleaning, we have 5,428 games remaining (detailed numbers of games are shown in Table 2).

Notice that **SPORTSSUM** (5,428 games) is much larger than the only public sports game summariza-

<sup>5</sup><https://match.sports.sina.com.cn/>

tion dataset (150 games) (Zhang et al., 2016).

**Statistics and properties.** Table 3 shows the statistics of SPORTSSUM. On average, there are 193.77 sentences per commentary document and 23.80 sentences per news article. After applying word segmentation by *pyltp* tool<sup>6</sup>, the average numbers of words for commentary documents and news reports are 1825.63 and 427.98, respectively.

As mentioned in Section 1, commentary sentences and news sentences are in quite different writing styles. Commentary sentences are more colloquial and informal, while news sentences are more narrative and well-organized. Also, commentaries contain a large number of player names, which makes the model easy to generate news reports with incorrect facts, as shown in Section 3.

### 3 Sports Game Summarization

The goal of sports game summarization is to generate a sports news report  $\tilde{R} = \{\tilde{r}_1, \tilde{r}_2, \dots, \tilde{r}_n\}$  from a given live commentary document  $C = \{(t_1, s_1, c_1), \dots, (t_m, s_m, c_m)\}$ . The generated news report  $\tilde{R}$  is expected to cover most of the important events in the games and describe those events correctly. In this paper, we propose a two-step model for SPORTSSUM. The proposed model first learns a *selector* to extract important commentary sentences and then utilizes a *rewriter* to convert the selected sentences to a news article.

**Sentence mapping.** To train the selector and rewriter, we need some labels to indicate the importance of commentary sentences and the corresponding news sentences. To obtain the labels, we consider the timeline information and BERTScore (Zhang et al., 2020), a metric to measure the sentence similarity, and map each news sentence to a commentary sentence. Although we have no explicit timeline information for news sentences, we observe that many news sentences start with “in the  $n$ -th minute” and thus we can extract the timeline information for some news sentences.

We map sentences by the following steps: 1) For each news sentence  $r_i$ , we extract the timeline information  $h_i$  if possible. Otherwise, we do not map this news sentence. 2) We consider those commentary sentences  $c_j$  with  $t_j$  being close to  $h_i$ . More specifically, we consider  $C^{(i)} = \{c_k, c_{k+1}, \dots, c_{k+l}\}$ , where  $c_j$  is the commentary sentence with timeline information  $t_j \in [h_i, h_i + 3]$

<sup>6</sup><https://github.com/HIT-SCIR/pyltp>

for  $k \leq j \leq k + l$ . 3) We compute BERTScore of the news sentence  $r_i$  and all the commentary sentences in  $C^{(i)}$ . The commentary sentence  $c_j \in C^{(i)}$  with the highest score is considered to be mapped with the news sentences  $r_i$ .

With the above mapping process, we obtain a set of mapped commentary sentences and news sentences  $\mathcal{D} = \{(\bar{c}_1, \bar{r}_1), (\bar{c}_2, \bar{r}_2), \dots, (\bar{c}_s, \bar{r}_s)\}$ , which can be used for training our selector and rewriter.

**Selector.** There are many commentary sentences in a live commentary document, but only few of them contain valuable information and should be reported in the news article. Therefore, we learn a *selector* to pick up those important sentences. More specifically, Given a commentary document  $C = \{(t_1, s_1, c_1), \dots, (t_m, s_m, c_m)\}$ , the selector outputs a set  $C_{select} = \{\tilde{c}_1, \tilde{c}_2, \dots, \tilde{c}_n\}$  which contains only important commentary sentences.

We train a binary classifier as the selector to choose important commentary sentences. When training, for each commentary sentence  $c_i$  in  $C$ , we assign a positive label if  $c_i$  can be mapped with a news sentence by the aforementioned mapping process. Otherwise, we give a negative label.

**Rewriter.** The rewriter converts the selected commentary sentences  $C_{select} = \{\tilde{c}_1, \tilde{c}_2, \dots, \tilde{c}_n\}$  to a news report  $\tilde{R} = \{\tilde{r}_1, \tilde{r}_2, \dots, \tilde{r}_n\}$ . We focus on the sentence-level rewriter. That is, we convert each selected commentary sentence  $\tilde{c}_i$  to a news sentence  $\tilde{r}_i$ . An intuitive way to learn the sentence-level rewriter is training a sequence-to-sequence (seq2seq) model, such as LSTM (Hochreiter and Schmidhuber, 1997) and Transformer (Vaswani et al., 2017), on the mapped sentences  $\mathcal{D}$ . However, as illustrated in Table 4, we observe that the seq2seq model tends to generate high-frequency player names rather than the correct player names even though the high-frequency player names do not appear in the commentary sentences. We call this situation *name mismatch problem*.

To solve the name mismatch problem, we train the rewriter in a *template-to-template* (tem2tem) way instead of in a seq2seq way. We first build a dictionary of player names from the lineup data (metadata). Next, for each  $(\bar{c}_i, \bar{r}_i)$  in  $\mathcal{D}$ , we replace all the player names in  $\bar{c}_i$  and  $\bar{r}_i$  with a special token “[player]” so that the new sentence is like a template. If there are multiple player names in a sentence, we append a number to the special token to distinguish them, as shown in Table 5.

Live Commentary Sentence	<b>里贝里</b> 禁区左侧尝试右脚射门,皮球高出球门.给他传球的是 <b>拉姆</b> . <b>Ribery</b> tried to shoot with his right foot from the left side of the penalty area, but the ball was higher than the crossbar. <b>Lahm</b> passed the ball to him.
Ground Truth News Sentence	<b>拉姆</b> 转移到左侧, <b>里贝里</b> 突入禁区左侧距门12米处抽射高出。 <b>Lahm</b> passed the ball to the left, and <b>Ribery</b> cut in the left penalty area and shot from 12 meters ahead the goal line. The shot was too high.
News Sentence Generated by Seq2seq Model	<b>里贝里</b> 传球, <b>曼朱基奇</b> 禁区左侧射门偏出远门柱。 <b>Ribery</b> passed the ball and <b>Mandzukic</b> 's shot from the left side of the penalty area was out of the goalpost.

Table 4: An example of the name mismatch problem. Seq2seq model tends to generate high-frequency player names rather than the correct names.

	Input Sentence	Output Sentence
Seq2seq	射门!!!里贝里球门线跟前右脚射门,被阿德勒横身扑出.给他传球的是 <b>拉姆</b> . Shoot!!! Ribery's right foot shot in front of the goal line was saved by Adler. Lahm passed the ball to him.	拉姆右路低传, 里贝里前点铲射被阿德勒封出。 Lahm made a low pass on the right and Ribery's shot from the front was blocked by Adler.
	射门!!![player1] 球门线跟前右脚射门,被[player2]横身扑出.给他传球的是[player3]. Shoot!!! [player1]'s right foot shot in front of the goal line was saved by [player2]. [player3] passed the ball to him.	[player3] 右路低传, [player1]前点铲射被[player2]封出。 [player3] made a low pass on the right and [player1]'s shot from the front was blocked by [player2].

Table 5: Training models by seq2seq versus training models by tem2tem.

After converting  $\bar{c}_i$  and  $\bar{r}_i$  to the template sentences, we train a seq2seq model on the template sentences. By training models in a tem2tem way, the model focuses more on the relations between players and actions and is less influenced by the high-frequency player names.

When predicting, for each commentary sentence  $\tilde{c}_i$  in  $C_{select}$ , we use the aforementioned way to convert  $\tilde{c}_i$  to a commentary template sentence. Then, we generate a news template sentence by the rewriter and replace all the special tokens in the sentence with the original player names.

## 4 Experiments

SPORTSSUM contains 5,428 games and we split them into three sets: training (4,828 games), validation (300 games), and testing (300 games) sets.

**Evaluation.** We consider ROUGE scores (Lin, 2004), which are standard metrics for summarization tasks. More precisely, we focus on ROUGE-1, ROUGE-2, and ROUGE-L. However, we observe that ROUGE scores cannot accurately evaluate the correctness of summaries. Some summaries may get high ROUGE scores but contain many incorrect facts. Therefore, we design two metrics: *name matching score* (NMS) and *event matching score* (EMS).

The name matching score evaluates the closeness of the player names in the ground truth news article  $R$  and the generated summaries  $\tilde{R}$ . Let  $N_g$  and  $N_p$  denote the set of the player names appearing in  $R$  and  $\tilde{R}$ , respectively. We define the name matching score as

$$\text{NMS}(R, \tilde{R}) = \text{F-score}(N_g, N_p).$$

Similarly, the event matching score evaluates the closeness of the events in  $R$  and  $\tilde{R}$ . We define an event as a pair (subject, verb) in the sentence. Two pairs ( $\text{subject}_1, \text{verb}_1$ ) and ( $\text{subject}_2, \text{verb}_2$ ) are viewed as equivalent if and only if **1**)  $\text{subject}_1$  is the same as  $\text{subject}_2$  and **2**)  $\text{verb}_1$  and  $\text{verb}_2$  are synonym<sup>7</sup> to each other. Let  $E_g$  and  $E_p$  represent the set of events in  $R$  and  $\tilde{R}$ , respectively, the event matching score is defined as

$$\text{EMS}(R, \tilde{R}) = \text{F-score}(E_g, E_p).$$

**Implementations and Models.** We consider the convolutional neural network (Kim, 2014) as the selector. For the rewriter, we consider the following: (1) **LSTM**: a bidirectional LSTM with attention mechanism (Bahdanau et al., 2015). (2) **Transformer**. (Vaswani et al., 2017) (3) **PGNet**: pointer-generator network, an encoder-decoder model with copy mechanism (See et al., 2017).

<sup>7</sup>Details to decide synonyms can be found in Appendix B.

Method	Model	ROUGE-1	ROUGE-2	ROUGE-L	NMS	EMS
Extractive Models	RawSent	26.52	7.64	25.42	57.33	36.17
	LTR	24.44	6.39	23.19	51.63	29.03
Abstractive Models	Abs-LSTM	30.54	10.16	29.78	10.87	14.03
	Abs-PGNet	34.02	11.09	33.13	17.87	19.76
Selector + Rewriter (Seq2seq)	LSTM	41.39	16.99	40.53	28.48	25.19
	Transformer	41.71	18.10	40.96	35.63	30.94
	PGNet	<b>43.17</b>	<b>18.66</b>	<b>42.27</b>	48.18	36.94
Selector + Rewriter (Tem2tem)	LSTM	41.71	17.08	40.82	<b>59.54</b>	40.34
	Transformer	41.47	17.18	40.54	58.26	39.33
	PGNet	41.95	17.09	41.01	59.35	<b>40.46</b>

Table 6: Evaluation results. NMS and EMS represent the name matching score and the event matching score.

For comparison, we consider two extractive summarization baselines: (1) **RawSent**: the raw sentences selected by the selector without rewriting. (2) **LTR**: the learning-to-rank approach for sports game summarization proposed by the previous work (Zhang et al., 2016).

In addition, we train a bidirectional LSTM with attention mechanism (**Abs-LSTM**) and a pointer-generator network (**Abs-PGNet**) on the paired commentaries and news articles as two simple abstractive summarization baselines. More implementation details can be found in Appendix C.

**Results.** Table 6 shows the experimental results. We observe that the extractive models (RawSent and LTR) get low ROUGE scores but high NMS and EMS. That means the extractive models can generate summaries with correct information, but the writing style is different from the ground truth. On the contrary, the abstractive models get higher ROUGE scores but lower NMS and EMS. That implies the summaries generated by the abstractive models usually contain incorrect facts.

Our proposed two-step model performs better than the extractive models and the abstractive models on ROUGE scores, NMS, and EMS. This verifies our design of the selector and the rewriter. In addition, we observe that when training the model in a tem2tem way, we can get better NMS and EMS, which implies that training by tem2tem can improve the correctness of summaries.

## 5 Related Work

**Text summarization.** Existing approaches can be grouped into two families: extractive models and abstractive models. Extractive models select a part of sentences from the source document as the summary. Traditional approaches (Carbonell and Goldstein, 1998; Erkan and Radev, 2004; Mc-

Donald, 2007) utilize graph or optimization techniques. Recently, neural models achieve good performance (Cheng and Lapata, 2016; Nallapati et al., 2017; Jadhav and Rajan, 2018). Abstractive summarization models aim to rephrase the source document. Most work applies neural models for this task. (Rush et al., 2015; Chopra et al., 2016; Nallapati et al., 2016; Zeng et al., 2016; See et al., 2017; Gehrmann et al., 2018).

**Factual correctness of summaries.** There is a lot of work focusing on evaluation and improvement of the factual correctness of summaries (Falke et al., 2019; Kryscinski et al., 2019; Wang et al., 2020; Maynez et al., 2020; Zhu et al., 2020).

**Data-to-Text generation.** Recently, generating news articles from different kinds of data-records becomes a popular research direction. Wiseman et al. (2017); Puduppully et al. (2019) focus on generating news from boxed-data. Zhang et al. (2016) and Yao et al. (2017) study generating sports news from live commentaries, but their methods are based on hand-crafted features.

## 6 Conclusion

We present SPORTSSUM, a Chinese dataset for sports game summarization, as well as a model that consists of a selector and a rewriter. To improve the quality of generated news, we train the model in a tem2tem way. We design two metrics to evaluate the correctness of generated summaries. The experimental results demonstrate that the proposed model performs well on ROUGE scores and the two designed scores.

## Acknowledgments

We thank Tecent AI Lab, UCLA-NLP group, and anonymous reviewers for their feedback.

## References

- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.
- Jaime G. Carbonell and Jade Goldstein. 1998. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *SIGIR*.
- Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *ACL*.
- Sumit Chopra, Michael Auli, and Alexander M. Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *NAACL*.
- Günes Erkan and Dragomir R. Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22:457–479.
- Tobias Falke, Leonardo F. R. Ribeiro, Prasetya Ajie Utama, Ido Dagan, and Iryna Gurevych. 2019. Ranking generated summaries by correctness: An interesting but challenging application for natural language inference. In *ACL*.
- Sebastian Gehrmann, Yuntian Deng, and Alexander M. Rush. 2018. Bottom-up abstractive summarization. In *EMNLP*.
- Karl Moritz Hermann, Tomás Kocišký, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *NIPS*.
- Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.
- Aishwarya Jadhav and Vaibhav Rajan. 2018. Extractive summarization with SWAP-NET: sentences and words from alternating pointer networks. In *ACL*.
- Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *EMNLP*.
- Wojciech Kryscinski, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Evaluating the factual consistency of abstractive text summarization. *Preprint arXiv:1910.12840*.
- Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*.
- Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan T. McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *ACL*.
- Ryan T. McDonald. 2007. A study of global inference algorithms in multi-document summarization. In *ECIR*.
- Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. SummaRuNNer: A recurrent neural network based sequence model for extractive summarization of documents. In *AAAI*.
- Ramesh Nallapati, Bowen Zhou, Cícero Nogueira dos Santos, Çağlar Gülcöhre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *CoNLL*.
- Ratish Puduppully, Li Dong, and Mirella Lapata. 2019. Data-to-text generation with content selection and planning. In *AAAI*.
- Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *EMNLP*.
- Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *ACL*.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*.
- Alex Wang, Kyunghyun Cho, and Mike Lewis. 2020. Asking and answering questions to evaluate the factual consistency of summaries. In *ACL*.
- Sam Wiseman, Stuart M. Shieber, and Alexander M. Rush. 2017. Challenges in data-to-document generation. In *EMNLP*.
- Jin-ge Yao, Jianmin Zhang, Xiaojun Wan, and Jianguo Xiao. 2017. Content selection for real-time sports news construction from commentary texts. In *INLG*.
- Wenyuan Zeng, Wenjie Luo, Sanja Fidler, and Raquel Urtasun. 2016. Efficient summarization with read-again and copy mechanism. *Preprint arXiv:1611.03382*.
- Jianmin Zhang, Jin-ge Yao, and Xiaojun Wan. 2016. Towards constructing sports news from live text commentary. In *ACL*.
- Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. BERTScore: Evaluating text generation with BERT. In *ICLR*.
- Chenguang Zhu, William Hinthon, Ruochen Xu, Qingkai Zeng, Michael Zeng, Xuedong Huang, and Meng Jiang. 2020. Boosting factual correctness of abstractive summarization with knowledge graph. *Preprint arXiv:2003.08612*.

## A Starting Keywords

We consider the following regular expressions as the starting keywords:

- 一开场
- 开场后
- 开场[\d]+分钟
- 开始[\d]+分钟
- 开场[仅][\d]+秒
- [\d]+秒
- 第[\d]+分钟
- [\d]+分钟
- [\d]+[米码]

## B Event Matching Score

We pick up the top 300 most frequent verbs and ask human to annotate if the verb is an important verb for soccer games or not. Then, we ask human to cluster those important verbs based on their meanings. When calculating the event matching score, we only consider those verbs. Two verbs are viewed as the synonym to each other if they are in the same group. The groups of verbs are as follows:

- **Shooting:** 射门, 打门, 攻门, 抽射, 推射, 劲射, 远射, 低射, 补射, 扫射, 斜射, 捅射, 射, 怒射, 起脚, 铲射, 垫射, 吊射, 挑射, 弹射, 勾射, 爆射, 头球, 甩头
- **Missed Shot:** 偏出, 高出, 打偏, 弹出, 打高, 弹回, 打飞, 顶高, 顶偏, 超出, 射偏, 蹤偏, 蹤出, 滑出
- **Passing:** 传中, 传球, 斜传, 送出, 头球摆渡, 直塞, 横传, 挑传, 直传, 低传, 横敲, 给到, 传入, 传, 传到, 妙传, 斜塞, 长传, 短传, 回传, 回敲, 回点, 分球
- **Blocking:** 扑出, 挡出, 没收, 封堵, 得到, 封出, 托出, 扑住, 救下, 抱住, 救出
- **Defense:** 解围, 破坏, 铲出, 化解
- **Foul:** 犯规, 吃到, 警告, 判罚, 被判, 领到, 罚下, 出示, 被罚

## C Implementation Details

For the selector, we consider CNN with the same architecture in (Kim, 2014) and set the learning rate to  $10^{-3}$ .

For the rewriter, the implementation details are as follows:

- **LSTM:** we use a bidirectional LSTM with the attention mechanism (Bahdanau et al., 2015). The size of hidden state is set to 300. We set the learning rate to  $10^{-3}$ .

- **Transformer:** we use the Transformer with the same architecture in the original paper (Vaswani et al., 2017). We set the learning rate to  $10^{-4}$ .

- **PGNet:** we implement the pointer-generator network (See et al., 2017) and set the size of hidden state to 300. We set the learning rate to  $10^{-3}$ .

- **LSTM-abs:** we use a bidirectional LSTM with the attention mechanism (Bahdanau et al., 2015). The size of hidden state is set to 300. We set the learning rate to  $10^{-3}$ .

- **PGNet-abs:** we implement the pointer-generator network (See et al., 2017) and set the size of hidden state to 300. We set the learning rate to  $10^{-3}$ .

For all the models, we use the 200-dimensional pre-trained Chinese word embedding from Tencent AI Lab<sup>8</sup>.

<sup>8</sup><https://ai.tencent.com/ailab/nlp/embedding.html>

# Massively Multilingual Document Alignment with Cross-lingual Sentence-Mover’s Distance

Ahmed El-Kishky

Facebook AI

ahe1k@fb.com

Francisco Guzmán

Facebook AI

fguzman@fb.com

## Abstract

Document alignment aims to identify pairs of documents in two distinct languages that are of comparable content or translations of each other. Such aligned data can be used for a variety of NLP tasks from training cross-lingual representations to mining parallel data for machine translation. In this paper we develop an unsupervised scoring function that leverages cross-lingual sentence embeddings to compute the semantic distance between documents in different languages. These semantic distances are then used to guide a document alignment algorithm to properly pair cross-lingual web documents across a variety of low, mid, and high-resource language pairs. Recognizing that our proposed scoring function and other state of the art methods are computationally intractable for long web documents, we utilize a more tractable greedy algorithm that performs comparably. We experimentally demonstrate that our distance metric performs better alignment than current baselines outperforming them by 7% on high-resource language pairs, 15% on mid-resource language pairs, and 22% on low-resource language pairs.

## 1 Introduction

While the Web provides a large amount of monolingual text, cross-lingual parallel data is more difficult to obtain. Despite its scarcity, parallel cross-lingual data plays a crucial role in a variety of tasks in natural language processing such as machine translation. Previous works have shown that training on sentences extracted from parallel or comparable documents mined from the Web can improve machine translation models (Munteanu and Marcu, 2005) or learning word-level translation lexicons (Fung and Yee, 1998; Rapp, 1999). Other tasks that leverage these parallel texts include cross-lingual information retrieval, document classification, and multilingual representations such as

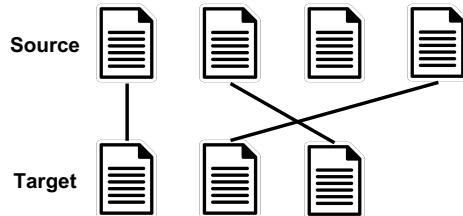


Figure 1: Documents in a source and target language in the same web-domain. Solid lines indicate cross-lingual document pairs.

XLM (Lample and Conneau, 2019). Document alignment is a method for obtaining cross-lingual parallel data that seeks to pair documents in different languages such that pairs are translations or near translations of each other. As seen in Figure 1, this involves a one-to-one pairing of documents in a source language with documents in a target language.

To automate and scale the process of identifying these documents pairs, we introduce an approach to accurately mine comparable web documents across a variety of low, mid, and high-resource language directions. Previous approaches have been applied to homogeneous corpora, however mining the Web involves analyzing a variety of heterogeneous data sources (Koehn et al., 2002). Other approaches rely on corpus-specific features such as metadata and publication date which can be inconsistent and unreliable (Munteanu and Marcu, 2005; AbduI-Rauf and Schwenk, 2009). Related methods utilize document structure when calculating document similarity (Resnik and Smith, 2003; Chen and Nie, 2000). However, when mining large, unstructured collections of web documents these features are often missing or unreliable. As such, we introduce an approach that aligns documents based solely on semantic distances between their textual content.

For our approach, we first decompose documents into sentences, and encode each sentence into a cross-lingual semantic space yielding a bag-

of-sentences representation. Utilizing the dense, cross-lingual representation of sentences, we then compute document distances using a variant of earth mover’s distance where probability mass is moved from the source document to the target document. We then leverage these document distances as a guiding metric for identifying cross-lingual document pairs and demonstrate experimentally that our proposed method outperforms state-of-the-art baselines that utilize cross-lingual document representations.

## 2 Related Works

Crawling and mining the web for parallel data has been previously explored by Resnik (1999) where the focus is on identifying parallel text from multilingual data obtained from a single source. For example, parallel corpora were curated from the United Nations General Assembly Resolutions (Rafalovitch et al., 2009; Ziemska et al., 2016) and from the European Parliament (Koehn, 2005). However, curating from homogeneous sources by deriving domain-specific rules does not generalize to arbitrary web-domains.

Other approaches rely on metadata for mining parallel documents in unstructured web corpora. Some methods leveraged publication date and other temporal heuristics to identifying parallel documents (Munteanu and Marcu, 2005, 2006; Udupa et al., 2009; Do et al., 2009; Abdul-Rauf and Schwenk, 2009). However, temporal features are often sparse, noisy, and unreliable. Another class of alignment methods rely on document structure (Resnik and Smith, 2003; Chen and Nie, 2000) yet these structure signals can be sparse and may not generalize to new domains.

In the WMT-2016 bilingual document alignment shared task (Buck and Koehn, 2016a), many techniques were proposed to retrieve, score, and align cross-lingual document pairs. However this shared task only considered English to French – a high-resource direction and the proposed techniques were not readily extendable to more languages.

Several approaches translate the target corpus into the source language, then apply retrieval and matching approaches on translated 2-grams and 5-grams to query, retrieve, and align documents (Dara and Lin, 2016; Gomes and Lopes, 2016). These methods rely on high-quality translation systems to translate, however such models may not exist, especially for low-resource language directions. Ad-

ditionally, these methods leverage rare n-grams to identify likely candidates, yet low-frequency words and phrases that are likely to be mistranslated by machine translation systems.

In the shared task, many document similarity measures were investigated for use in aligning English to French web documents. One method utilized a phrase table from a phrase-based statistical machine translation system to compute coverage scores, based on the ratio of phrase pairs covered by a document pair (Gomes and Lopes, 2016). Other methods utilize the translated content of the target (French) document, and find the source (English) corresponding document based on n-gram matches in conjunction with a heuristic document length ratio (Dara and Lin, 2016; Shchukin et al., 2016). Other methods translate the target documents into the source language and apply cosine similarity between tf/idf weighted vectors on unigrams and n-grams (Buck and Koehn, 2016b; Medved et al., 2016; Jakubina and Langlais, 2016). Finally, several methods were introduced that score pairs using metadata in each document such as links to documents, URLs, digits, and HTML structure (Esplà-Gomis et al., 2016; Papavassiliou et al., 2016).

Recently, the use of neural embedding methods has been explored for bilingual alignment of text at the sentence and document level. One method proposes using hierarchical document embeddings, constructed from sentence embeddings, for bilingual document alignment (Guo et al., 2019). Another method leverages a multilingual sentence encoder to embed individual sentences from each document, then performs a simple vector average across all sentence embeddings to form a dense document representation with cosine similarity guiding document alignment (El-Kishky et al., 2019).

Word mover’s distance (WMD) is an adaptation of earth mover’s distance (EMD) (Rubner et al., 1998) that has been recently used for document similarity and classification (Kusner et al., 2015; Huang et al., 2016; Atasu et al., 2017). Other methods have leveraged the distance for cross-lingual document retrieval (Balikas et al., 2018). However these methods treat individual words as the base semantic unit for comparison which are intractable for large web-document alignment.

Finally, sentence mover’s similarity has been proposed for automatically evaluating machine-generated texts outperforming ROUGE (Clark et al., 2019). This method is purely monolingual

and sentence representations are constructed by summing individual word embeddings.

### 3 Problem Definition

Given a set of source documents,  $D_s$  and a set of target documents  $D_t$ , there exist  $|D_s| \times |D_t|$  potential pairs of documents of the form  $(d_s, d_t)$ . Let  $\mathcal{P}$  be the set of all candidate pairs  $(D_s \times D_t)$ . Then cross-lingual document alignment aims to find the largest mapping from source documents to target documents,  $\mathcal{P}' \subset \mathcal{P}$ , s.t. given an  $D_s$  and  $D_t$  where, without a loss of generality,  $|D_s| \leq |D_t|$ , the largest *injective function mapping* between  $D_s$  and  $D_t$ :

$$\forall a, b \in D_s, (a, c) \in \mathcal{P}' \wedge (b, c) \in \mathcal{P}' \implies a = b$$

In other words, each source document and target document can only be used in at most a single pair. This can be seen in Figure 1 where within the same web-domain, given source and target documents, the task is to match each source document to a unique target document where possible.

To find the best possible mapping between  $D_s$  and  $D_t$  we require two components: 1) a similarity function  $\phi(d_s, d_t)$  which is used to score a set of candidate document pairs according to their semantic relatedness; and 2) an alignment or matching algorithm which uses the scores for each of the pairs in  $D_s \times D_t$  to produce an alignment of size  $\min(|D_s|, |D_t|)$  representing the best mapping according to  $\phi(d_s, d_t)$ .

## 4 Cross-Lingual Sentence Mover’s Distance

WMD fails to generalize to our use case for two reasons: (1) it relies on monolingual word representations which fail to capture the semantic distances between different language documents (2) intractability due to long web documents or lack word boundaries in certain languages.

To address this, we introduce cross-lingual sentence mover’s distance (SMD) and show that representing each document as a bag-of-sentences (BOS) and leveraging recent improvements in multilingual sentence representations, SMD can better identify cross-lingual document pairs.

### 4.1 Cross-Lingual Sentence Mover’s Distance

Our proposed SMD solves the same optimization problem as WMD, but utilizes cross-lingual sentence embeddings instead of word embeddings as

the base semantic. In particular, we utilize LASER sentence representations (Artetxe and Schwenk, 2019). LASER learns to simultaneously embed 93 languages covering 23 different alphabets into a joint embedding space by training a sequence-to-sequence system on many language pairs at once using a shared encoder and a shared byte-pair encoding (BPE) vocabulary for all languages. Utilizing LASER, each sentence is encoded using an LSTM encoder into a fixed-length dense representation.

We adapt EMD to measure the distance between two documents by comparing the distributions of sentences within each document. More specifically, SMD represents each document as a *normalized bag-of-sentences* (nBOS) where each sentence has associated with it some probability mass. As distances can be computed between dense sentence embeddings, the overall document distance can then be computed by examining how close the distribution of sentences in the source document is to sentences in the target document. We formulate this distance as the minimum cost of transforming one document into the other.

For our basic formulation of SMD, each document is represented by the relative frequencies of sentences, i.e., for the  $i_{th}$  sentence in the document,

$$d_{A,i} = \text{cnt}(i)/|A| \quad (1)$$

where  $|A|$  is the total number of sentence in document A, and  $d_{B,i}$  is defined similarly for document B. Under this assumption, each individual sentence in a document is equally important and probability mass is allocated uniformly to each sentence. Later, we will investigate alternative schemes to allocating probability mass to sentences.

Now let the  $i_{th}$  sentence be represented by a vector  $v_i \in R^m$ . This length-m dense embedding representation for each sentence allows us to define distances between the  $i_{th}$  and  $j_{th}$  sentences. We denote  $\Delta(i, j)$  as the distance between the  $i_{th}$  and  $j_{th}$  sentences and let  $V$  denote the vocabulary size where the vocabulary is the unique set of sentences within a document pair. We follow previous works (Kusner et al., 2015) and use the Euclidean distance,  $\Delta(i, j) = \|v_i - v_j\|$ . The SMD between a document pair is then the solution to the linear program:

$$SMD(A, B) = \min_{T \geq 0} \sum_{i=1}^V \sum_{j=1}^V T_{i,j} \times \Delta(i, j) \quad (2)$$

subject to:

$$\forall i \sum_{j=1}^V T_{i,j} = d_{A,i}$$

$$\forall j \sum_{i=1}^V T_{i,j} = d_{B,j}$$

Where  $T \in R^{V \times V}$  is a nonnegative matrix, where each  $T_{i,j}$  denotes how much of sentence  $i$  in document  $A$  is assigned to sentences  $j$  in document  $B$ , and constraints ensure the flow of a given sentence cannot exceed its allocated mass. Specifically, SMD ensures the entire outgoing flow from sentence  $i$  equals  $d_{A,i}$ , i.e.  $\sum_j T_{i,j} = d_{A,i}$ . Additionally, the amount of incoming flow to sentence  $j$  must match  $d_{B,j}$ , i.e.,  $\sum_i T_{i,j} = d_{B,j}$ .

## 4.2 Alternative Sentence Weighting Schemes

In Equation 1, each document is represented as a normalized bag-of-sentences (nBOS) where sentences are equally weighted. However, we posit that some sentences may be more semantically important than others.

**Sentence Length Weighting** The first insight we investigate is that documents will naturally be segmented into sentences of different lengths based on the language, content, and choice of segmentation. While Equation 1, treats each sentence equally, we posit that longer sentences should be assigned larger weighting than shorter sentences.

As such, we weight each sentence by the number of tokens in the sentence relative to the total number of tokens in the entire document, i.e., for the  $i_{th}$  sentence in the document  $A$ , we compute the weighting  $SL(i)$  as follows:

$$d_{A,i} = cnt(i) \cdot |i| / \sum_{s \in A} cnt(s) \cdot |s| \quad (3)$$

where  $|i|$  and  $|s|$  indicate the number of tokens in sentences  $i$  and  $s$  respectively. As such, longer sentence receive larger probability mass than shorter sentences.

**IDF Weighting** The second insight we investigate is that text segments such as titles and navigation text is ubiquitous in crawled data yet less semantically informative. Based on this insight, we apply a variant of inverse document frequency (IDF) – a weighting scheme common in the information retrieval space – to individual sentences (Robertson, 2004). Under this scheme, the

more common a sentence is within a webdomain, the less mass the sentence will be allocated.

For sentence  $i$  in a web-domain  $D$ , we compute  $IDF(i)$  as follows:

$$d_{A,i} = 1 + \log \frac{|D|}{|\{d \in D : i \in d\}|} \quad (4)$$

where  $|\{d \in D : s \in d\}|$  is the number of documents where the sentence  $s$  occurs and smoothing by 1 is performed to prevent 0 IDF.

**SLIDF Weighting** Finally, we propose combining both sentence length and inverse document frequency into a joint weighting scheme:

$$d_{A,i} = SL(i) \cdot IDF(i) \quad (5)$$

In this scheme, each sentence is weighted proportionally to the number of tokens it contains as well as by the IDF of the sentence within the domain. This weighting scheme is reminiscent of the use of tf-idf to determine word relevance (Ramos et al., 2003), but instead sentence length and idf are used to determine sentence importance.

## 4.3 Fast Distance Approximation

While EMD and other variants have demonstrated superior performance in many retrieval and classification tasks, they have also been shown to suffer from high computational complexity  $\mathcal{O}(p^3 \log p)$ , where  $p$  denotes the number of unique semantic units in a document pair. As such, we investigate techniques to speed up this computation.

**Relaxed SMD** Given the scalability challenges for computing WMD, simplified version of WMD was proposed that relaxes one of the two constraints in the original formulation (Kusner et al., 2015). Applying the same principle to SMD, we formulate:

$$SMD(A, B) = \min_{T \geq 0} \sum_{i=1}^V \sum_{j=1}^V T_{i,j} \times \Delta(i, j)$$

subject to:  $\forall i \sum_{j=1}^V T_{i,j} = d_{A,i}$ . Analogous to the relaxed-WMD, this relaxed problem yields a lower-bound to the SMD as every SMD solution satisfying both constraints remains a feasible solution if one constraint is removed. The optimal solution can be found by simply allocating the mass in each source sentence to the closest sentence in the target document.

The same computation can be performed in the reverse direction by removing the second constraint:  $\forall j \sum_{i=1}^V T_{i,j} = d_{B,j}$ . Similarly, the optimal solution allocates the mass sentences in the target document to the closest sentence in the source document. Both these distances can be calculated by computing the distance matrix between all pairs of sentences in  $\mathcal{O}(p^2)$  time. For a tighter estimate of distance, the maximum of the two resultant distances can be used.

**Greedy Mover’s Distance** We introduce an alternative to the relaxed-EMD variant wherein we keep both constraints in the transportation problem, but identify an approximate transportation scheme. This greedy mover’s distance (GMD) finds the closest sentence pair between the source and target and moves as much mass between the two sentences as possible; the algorithm moves to the next closest until all mass has been moved while maintaining both constraints.

---

#### Algorithm 1: Greedy Mover’s Distance

---

**Input:**  $d_s, d_t, w_s, w_t$   
**Output:**  $\Delta(d_s, d_t)$

```

1 pairs  $\leftarrow \{(s_s, s_t) \text{ for } s_s, s_t \in d_s \times d_t\}$ 
    in ascending order by  $\|s_s - s_t\|$ 
2 distance  $\leftarrow 0.0$ 
3 for  $s_s, s_t \in \text{pairs}$  do
4   flow  $\leftarrow \min(w_s[s_s], w_t[s_t])$ 
5    $w_s[s_s] \leftarrow w_s[s_s] - \text{flow}$ 
6    $w_t[s_t] \leftarrow w_t[s_t] - \text{flow}$ 
7   distance  $\leftarrow \text{distance} + \|s_s - s_t\| \times \text{flow}$ 
8 end
9 return total

```

---

As seen in Algorithm 1, the algorithm takes a source document ( $d_s$ ) and a target document ( $d_t$ ) as well as the probability mass for the sentences in each: respectively  $w_s$  and  $w_t$ . The algorithm first computes the euclidean distance between each sentence pair from source to target and sorts these pairs in ascending order by their euclidean distance. The algorithm then iteratively chooses the closest sentence pair and moves the mass of the smallest sentence from the source to the target and subtracting this moved math from both. The algorithm terminates when all moveable mass has been moved. Unlike the exact solution to EMD, the runtime complexity is a more tractable  $\mathcal{O}(|d_s||d_t| \times \log(|d_s||d_t|))$  which is dominated by the cost of sorting all candidate pairs. Unlike the relaxation, both constraints are satisfied but the transport is not necessarily optimal. As such, GMD

yields an upper-bound to the exact computation.

We experimentally compare the effect of both approximation strategies on downstream document alignment in Section 7.

## 5 Document Matching Algorithm

In addition to a distance metric (i.e. SMD), we need a document matching algorithm to determine the best mapping between documents in two languages.

In our case, this works as follows: for any given webdomain, each document in the source document set,  $D_s$  is paired with each document in the target set,  $D_t$ , yielding  $|D_s \times D_t|$  scored pairs – a fully connected bipartite graph representing all candidate pairings. Similar to previous works (Buck and Koehn, 2016b), the expected output assumes that each webpage in the non-dominant language has a translated or comparable counterpart. As visualized in Figure 1, this yields a  $\min(|D_s|, |D_t|)$  expected number of aligned pairs.

While an optimal matching maximizing scoring can be solved using the Hungarian algorithm (Munkres, 1957), the complexity of this algorithm is  $\mathcal{O}(\max(|D_s||D_t|)^3)$  which is intractable to even moderately sized web domains. As such, similar to the work in (Buck and Koehn, 2016b), a one-to-one matching between English and non-English documents is enforced by applying, competitive matching, a greedy bipartite matching algorithm.

---

#### Algorithm 2: Competitive Matching

---

**Input:**  $P = \{(d_s, d_t) | d_s \in D_s, d_t \in D_t\}$   
**Output:**  $P' = \{(d_{s,i}, d_{t,i}), \dots\} \subset P$

```

1 scored  $\leftarrow \{(p, \text{score}(p)) \text{ for } p \in P\}$ 
2 sorted  $\leftarrow \text{sort}(\text{scored})$  in ascending order
3 aligned  $\leftarrow \emptyset$ 
4  $S_s \leftarrow \emptyset$ 
5  $S_t \leftarrow \emptyset$ 
6 for  $d_s, d_t \in \text{sorted}$  do
7   if  $d_s \notin S_s \wedge d_t \notin S_t$  then
8     aligned  $\leftarrow \text{aligned} \cup \{(d_s, d_t)\}$ 
9      $S_s \leftarrow S_s \cup d_s$ 
10     $S_t \leftarrow S_t \cup d_t$ 
11 end
12 return aligned

```

---

In Algorithm 2, the algorithm first scores each candidate document pair using a distance function and then sorts pairs from closest to farthest. The algorithm then iteratively selects the closest document pair as long as the  $d_s$  and  $d_t$  of each pair have not been used in a previous (closer) pair. The

algorithm terminates when  $\min(|D_s|, |D_t|)$  pairs have been selected. Unlike the Hungarian algorithm, the runtime complexity is a more tractable  $\mathcal{O}(|D_s||D_t| \times \log(|D_s||D_t|))$  which is dominated by the cost of sorting all candidate pairs.

## 6 Experiments and Results

In this section, we explore the question of whether SMD can be used as a dissimilarity metric for the document alignment problem. Moreover, we explore which sentence weighting schemes yield the best results.

### 6.1 Experimental Setup

**Dataset** We evaluate on the test set from the URL-Aligned CommonCrawl dataset ([El-Kishky et al., 2019](#)) across 47 language directions.

**Baseline Methods** For comparison, we implemented two existing and intuitive document scoring baselines from ([El-Kishky et al., 2019](#)). The direct embedding (DE), directly embeds the entire content of a document using LASER. The second method sentence averaging (SA) embeds all sentences in a document using LASER and averages all embeddings to get a document representation. Cosine similarity on the embedded representation is used to compare documents.

**SMD Weightings** We evaluate four weighting schemes for SMD: (1) vanilla SMD with each sentence equally weighted(2) weighting by sentence length (SL) where SMD is computed under a scheme where each sentence is weighted by its length (number of tokens) normalized by the length of the entire document (3) weighting by inverse document frequency (IDF) where SMD is computed under a scheme where each sentence is weighted by the idf of the sentence (4) computing SMD under a scheme where each sentence is weighted by both sentence length and inverse document frequency (SLIDF). Under all these schemes, all weights are normalized to unit measure.

**Distance approximation** We use the greedy mover’s distance approximation for all variants reported. In Section 7 we further explore the performance of the full distance computation and relaxed variants that were described in Section 4.3.

### Evaluation Metric for Document Alignment

Because the ground-truth document pairs only reflect a high-precision set of web-document pairs

that are translations or of comparable content, there may be many other valid cross-lingual document pairs within each web-domain that are not included in the ground truth set. As such, we evaluate each method’s generated document pairs solely on the recall (i.e. what percentage of the aligned pages in the test set are found) from the ground truth pairs.

For each scoring method, we score document pairs from the source and target languages within the same web-domain using the proposed document distance metrics described above. For the alignment, we report the performance for each distance metric after applying the competitive matching alignment algorithm as described in Algorithm 2.

### 6.2 Results

In Table 1, we first notice that constructing document representations by directly embedding (DE) the entire content of each document and computing document similarity using cosine similarity of the representation severely under-performs compared to individually embedding sentences and constructing the document representations by averaging the individual sentence representations within the document (SA). This is intuitive as LASER embeddings were trained on parallel sentences and embedding much larger documents directly using LASER results in poorer representations than by first embedding smaller sentences and combining them into the final document representation.

Comparing the basic SMD to the best performing baseline (SA), we see a 4%, 12%, and 20% improvement across high, mid, and low-resource directions respectively. This improvement suggests that summing sentence embeddings into a single document representation degrades the quality of the resultant document distances over computing document distances by keeping all sentence representations separate and computing distances between individual sentence pairs and combining these distances into a final document distance. This is more pronounced in lower-resource over higher-resource pairs which may be due to poorer lower-resource embeddings due to LASER being trained on fewer low-resource sentence pairs. As such averaging is more destructive to these representations while SMD avoids this degradation.

Further analysis verified the intuition that different sentences should be allocated different weighting in SMD. Assigning mass proportional to the

Language	Recall						Language	Recall						Language	Recall					
	DE	SA	SMD	SL	IDF	SLIDF		DE	SA	SMD	SL	IDF	SLIDF		DE	SA	SMD	SL	IDF	SLIDF
French	0.39	0.84	0.81	0.84	0.83	<b>0.85</b>	Romanian	0.15	0.40	0.44	0.43	<b>0.45</b>	0.43	Estonian	0.28	0.52	0.69	0.66	<b>0.74</b>	0.72
Spanish	0.34	0.53	0.59	0.63	0.62	<b>0.64</b>	Vietnamese	0.06	0.28	0.29	0.29	<b>0.32</b>	Bengali	0.05	0.32	0.78	0.72	0.77	<b>0.79</b>	
Russian	0.06	0.64	0.69	0.69	0.70	<b>0.71</b>	Ukrainian	0.05	0.68	0.67	0.78	0.78	<b>0.82</b>	Albanian	0.23	0.56	<b>0.66</b>	0.65	0.65	<b>0.66</b>
German	0.52	0.74	<b>0.78</b>	0.76	0.77	0.77	Greek	0.05	0.31	0.47	0.48	<b>0.49</b>	<b>0.49</b>	Macedonian	0.02	0.33	0.32	0.36	<b>0.38</b>	0.33
Italian	0.22	0.47	0.55	0.56	0.56	<b>0.59</b>	Korean	0.06	0.34	0.60	0.54	<b>0.61</b>	0.60	Urdu	0.06	0.22	<b>0.60</b>	<b>0.60</b>	0.49	0.56
Portuguese	0.17	0.36	0.39	<b>0.41</b>	0.38	0.40	Arabic	0.04	0.32	0.63	0.59	<b>0.65</b>	0.61	Serbian	0.06	0.59	<b>0.75</b>	0.74	0.74	0.71
Dutch	0.28	0.49	0.54	0.54	0.54	<b>0.56</b>	Croatian	0.16	0.37	0.40	0.40	<b>0.41</b>	0.40	Azerbaijani	0.08	0.34	0.74	0.74	<b>0.75</b>	0.74
Indonesian	0.11	0.47	0.49	0.52	0.51	<b>0.53</b>	Slovak	0.20	0.41	0.46	<b>0.46</b>	<b>0.46</b>	0.44	Armenian	0.02	0.18	0.32	0.35	0.34	<b>0.38</b>
Polish	0.17	0.38	0.45	0.45	<b>0.46</b>	<b>0.46</b>	Thai	0.02	0.19	0.41	0.33	<b>0.47</b>	0.41	Belarusian	0.07	0.47	0.67	0.69	<b>0.73</b>	0.71
Turkish	0.12	0.38	0.52	0.56	0.57	<b>0.59</b>	Hebrew	0.05	0.18	0.39	<b>0.43</b>	0.41	0.41	Georgian	0.06	0.24	0.46	<b>0.48</b>	0.45	0.45
Swedish	0.19	0.40	0.44	0.44	<b>0.46</b>	0.45	Hindi	0.04	0.27	0.34	<b>0.54</b>	0.52	0.53	Tamil	0.02	0.20	0.51	0.45	0.51	<b>0.53</b>
Danish	0.27	0.62	0.63	<b>0.69</b>	0.65	<b>0.69</b>	Hungarian	0.15	0.49	0.50	<b>0.54</b>	0.51	<b>0.54</b>	Marathi	0.02	0.11	0.43	<b>0.46</b>	0.33	0.39
Czech	0.15	0.40	0.43	<b>0.44</b>	<b>0.44</b>	0.43	Lithuanian	0.11	0.73	0.79	0.79	<b>0.80</b>	<b>0.80</b>	Kazakh	0.05	0.31	0.44	<b>0.46</b>	0.45	0.45
Bulgarian	0.07	0.43	0.52	0.54	<b>0.55</b>	0.52	Slovenian	0.13	0.33	0.34	0.35	<b>0.36</b>	<b>0.36</b>	Mongolian	0.03	0.13	0.18	0.22	0.21	<b>0.23</b>
Finnish	0.06	0.47	0.51	0.51	<b>0.54</b>	0.52	Persian	0.06	0.32	0.56	0.57	0.53	<b>0.59</b>	Burmese	0.01	0.10	0.26	0.33	<b>0.46</b>	<b>0.46</b>
Norwegian	0.13	0.33	0.37	0.39	<b>0.42</b>	0.41							Bosnian	0.18	0.64	0.61	0.69	0.65	<b>0.72</b>	
AVG	0.20	0.50	0.54	0.56	0.56	<b>0.57</b>	AVG	0.09	0.37	0.49	0.50	<b>0.52</b>	<b>0.52</b>	AVG	0.08	0.33	0.53	0.54	0.54	<b>0.55</b>

(a) High-resource directions.

(b) Mid-resource directions.

(c) Low-resource directions.

Table 1: Alignment recall on URL-aligned CommonCrawl dataset.

number of tokens in the sentence (SL), we see a 2%, 1% and 1% absolute improvement in recall in high, mid, and low-resource directions over assigning equal probability mass. This supports the claim that longer sentences should be allocated higher importance weight over shorter sentences as they contain more semantic content. The second assumption we investigated is that sentences that are common within a webdomain have less semantic importance and should be allocated less probability mass when computing SMD. After computing SMD with each sentence allocated mass according to inverse document frequency (IDF) and normalized to unit measure, we see a 2%, 3%, and 1% improvement over SMD for high, mid, and low-resource directions. Finally, when combining both sentence length and inverse document frequency (SLIDF) and normalizing to unit measure, we see a 3%, 3% and 2% absolute improvement in recall for high, mid, and low-resource directions. Overall, our SMD with SLIDF weighting scheme outperforms the sentence averaging baseline by 7% on high-resource directions, 15% on mid-resource directions, and 22% on low-resource directions.

## 7 Discussion

Although using sentences over words as the base semantic unit drastically reduces the overall cost of computing EMD-based metrics, the cubic computation still prohibits its use as a fast distance metric for large-scale alignment efforts. As such, in Section 4.3 we described two faster approximations to EMD computation: (1) a relaxation of constraints resulting in a lower bound and (2) a greedy algo-

rithm for computing assigning transport representing an upper bound. We first analyze and compare the distances from each approximation scheme to the exact SMD computation.

Method	Tau	Recall	MAE	Runtime (s)
Exact-SMD	1.00	0.69	0.000	0.402
Relaxed-SMD	0.70	0.58	0.084	0.031
Greedy-SMD	0.98	0.69	0.010	0.107

Table 2: Comparing exact SMD computation to approximation schemes for computing SMD on 10 webdomains.

In Figure 3, we see that the distance computations for exact SMD and the greedy SMD approximation are highly correlated with small variance, while the relaxed approximation is less so with high variance. Additionally, as discussed in Section 4.3, the visualizations empirically suggest that our greedy approximation is a fairly tight upper bound while the relaxed approximation is a looser lower bound.

In Table 2, we compare quantitative metrics for the relaxed and greedy approximations to the exact solution of SMD on ten webdomains. Our first evaluation investigates how the approximate computation of distances affects the resultant ordering of document pairs. For the ten selected webdomains, we sort the document pairs in order by their computed distances and compare the ordering to the ordering induced by the exact computation of SMD. We evaluate the orderings using the Kendall-Tau metric (Kendall, 1938) which measures the agreement between the two rankings; if the agreement between the two rankings is perfect (i.e., the two

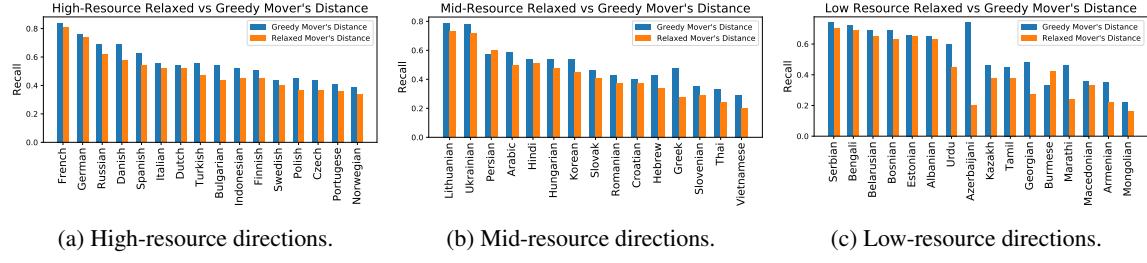


Figure 2: Document alignment results for different distance approximation techniques.

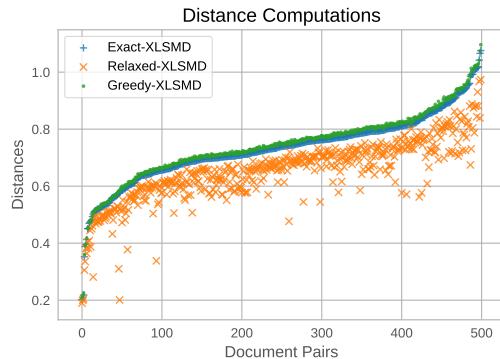


Figure 3: Exact, relaxed, and greedy-SMD distances sorted by Exact-SMD for a random selection of document pairs.

rankings are the same) the coefficient has value 1 and if the disagreement between the two rankings is perfect (i.e., one ranking is the reverse of the other) the coefficient has value -1. Intuitively, we would like the distances computed by an approximation to induce a similar ordering to the ordering by the exact distance computation. Comparing the Kendall-Tau for the relaxed and greedy approximations in relation to the exact computation shows that the order induced by the greedy approximation is very similar to the ordering induced by the exact computation while the relaxed approximation varies considerably. Additionally, the relaxed approximation demonstrates fairly high mean absolute error (MAE) and results in lower document alignment recall when compared to the exact computation of SMD, while our greedy approximation performs comparably and shows insignificant MAE. Finally, while the runtime of the relaxed computation is the fastest at 13 times faster than the exact computation, our greedy algorithm is approximately 4 times faster while delivering comparable document alignment performance to the exact computation and superior performance to the relaxed computation.

To ensure that the greedy algorithm consistently outperforms the relaxed algorithm on document alignment, we investigate the effect of using each

approximation method on the downstream document alignment performance across 47 language pairs of varying resource availability.

Approximation	Low	Mid	High	All
Relaxed-SMD	0.44	0.43	0.50	0.46
Greedy-SMD	0.54	0.50	0.56	0.54

Table 3: Document alignment performance of fast methods for approximating the same variant of SMD.

As seen in Figure 2, in 45 of the 47 evaluated language pairs, our proposed Greedy Mover’s Distance approximation yielded higher downstream recall in our alignment task over using the relaxed distance proposed for use in WMD (Kusner et al., 2015). In Table 3, we see a 10%, 7%, and 6% improvement in downstream recall across low, mid, and high-resource directions respectively. These results indicate that relaxing one of the two constraints in EMD is too lax for measuring an accurate distance. We posit this is because there are many sentences that can be considered “hubs” that are semantically close to many other sentences. These sentences can have a lot of probability mass allocated to them, resulting in a lower approximate EMD. Our greedy approximation ensures that both constraints are maintained even if the final result does not reflect the optimal transport.

## 8 Conclusion

In this paper, we introduce SMD a cross-lingual sentence mover’s distance metric for automatically assessing the semantic similarity of two documents in different languages. We leverage state-of-the-art multilingual sentence embeddings and apply SMD to the task of cross-lingual document alignment. We demonstrate that our new metric outperforms other unsupervised metrics by a margin, especially in medium and low-resourced conditions.

## References

- Sadaf Abdul-Rauf and Holger Schwenk. 2009. On the use of comparable corpora to improve smt performance. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 16–23. Association for Computational Linguistics.
- Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.
- Kubilay Atasu, Thomas Parnell, Celestine Dünner, Manolis Sifalakis, Haralampos Pozidis, Vasileios Vasileiadis, Michail Vlachos, Cesar Berrospi, and Abdel Labbi. 2017. Linear-complexity relaxed word mover’s distance with gpu acceleration. In *2017 IEEE International Conference on Big Data (Big Data)*, pages 889–896. IEEE.
- Georgios Balikas, Charlotte Laclau, Ievgen Redko, and Massih-Reza Amini. 2018. Cross-lingual document retrieval using regularized wasserstein distance. In *European Conference on Information Retrieval*, pages 398–410. Springer.
- Christian Buck and Philipp Koehn. 2016a. Findings of the wmt 2016 bilingual document alignment shared task. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 554–563.
- Christian Buck and Philipp Koehn. 2016b. Quick and reliable document alignment via tf/idf-weighted cosine distance. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 672–678.
- Jiang Chen and Jian-Yun Nie. 2000. Parallel web text mining for cross-language ir. In *Content-Based Multimedia Information Access-Volume 1*, pages 62–77. LE CENTRE DE HAUTES ETUDES INTERNATIONALES D’INFORMATIQUE DOCUMENTAIRE.
- Elizabeth Clark, Asli Celikyilmaz, and Noah A Smith. 2019. Sentence mover’s similarity: Automatic evaluation for multi-sentence texts. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2748–2760.
- Aswarth Abhilash Dara and Yiu-Chang Lin. 2016. Yoda system for wmt16 shared task: Bilingual document alignment. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 679–684.
- Thi-Ngoc-Diep Do, Viet-Bac Le, Brigitte Bigi, Laurent Besacier, and Eric Castelli. 2009. Mining a comparable text corpus for a vietnamese-french statistical machine translation system. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 165–172. Association for Computational Linguistics.
- Ahmed El-Kishky, Vishrav Chaudhary, Francisco Guzman, and Philipp Koehn. 2019. A massive collection of cross-lingual web-document pairs. *arXiv preprint arXiv:1911.06154*.
- Miquel Esplà-Gomis, Mikel Forcada, Sergio Ortiz Rojas, and Jorge Ferrández-Tordera. 2016. Bitextor’s participation in wmt’16: shared task on document alignment. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 685–691.
- Pascale Fung and Lo Yuen Yee. 1998. An ir approach for translating new words from nonparallel, comparable texts. In *COLING 1998 Volume 1: The 17th International Conference on Computational Linguistics*.
- Luís Gomes and Gabriel Pereira Lopes. 2016. First steps towards coverage-based document alignment. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 697–702.
- Mandy Guo, Yinfei Yang, Keith Stevens, Daniel Cer, Heming Ge, Yun-hsuan Sung, Brian Strope, and Ray Kurzweil. 2019. Hierarchical document encoder for parallel corpus mining. In *Proceedings of the Fourth Conference on Machine Translation*, pages 64–72, Florence, Italy. Association for Computational Linguistics.
- Gao Huang, Chuan Guo, Matt J Kusner, Yu Sun, Fei Sha, and Kilian Q Weinberger. 2016. Supervised word mover’s distance. In *Advances in Neural Information Processing Systems*, pages 4862–4870.
- Laurent Jakubina and Philippe Langlais. 2016. Bad luc@ wmt 2016: a bilingual document alignment platform based on lucene. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 703–709.
- Maurice G Kendall. 1938. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93.
- Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT summit*, volume 5, pages 79–86.
- Philipp Koehn et al. 2002. Europarl: A multilingual corpus for evaluation of machine translation.
- Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. 2015. From word embeddings to document distances. In *International conference on machine learning*, pages 957–966.
- Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*.

- Marek Medved̄, Miloš Jakubícek, and Vojtech Kovář. 2016. English-french document alignment based on keywords and statistical translation. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 728–732.
- James Munkres. 1957. Algorithms for the assignment and transportation problems. *Journal of the society for industrial and applied mathematics*, 5(1):32–38.
- Dragos Stefan Munteanu and Daniel Marcu. 2005. Improving machine translation performance by exploiting non-parallel corpora. *Computational Linguistics*, 31(4):477–504.
- Dragos Stefan Munteanu and Daniel Marcu. 2006. Extracting parallel sub-sentential fragments from non-parallel corpora. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 81–88. Association for Computational Linguistics.
- Vassilis Papavassiliou, Prokopis Prokopidis, and Stelios Piperidis. 2016. The ilsp/arc submission to the wmt 2016 bilingual document alignment shared task. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 733–739.
- Alexandre Rafalovitch, Robert Dale, et al. 2009. United nations general assembly resolutions: A six-language parallel corpus. In *Proceedings of Machine Translation Summit XII*.
- Juan Ramos et al. 2003. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 133–142. Piscataway, NJ.
- Reinhard Rapp. 1999. Automatic identification of word translations from unrelated english and german corpora. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 519–526. Association for Computational Linguistics.
- Philip Resnik. 1999. Mining the web for bilingual text. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 527–534. Association for Computational Linguistics.
- Philip Resnik and Noah A Smith. 2003. The web as a parallel corpus. *Computational Linguistics*, 29(3):349–380.
- Stephen Robertson. 2004. Understanding inverse document frequency: on theoretical arguments for idf. *Journal of documentation*, 60(5):503–520.
- Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. 1998. A metric for distributions with applications to image databases. In *Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)*, pages 59–66. IEEE.
- Vadim Shchukin, Dmitry Khristich, and Irina Galinskaya. 2016. Word clustering approach to bilingual document alignment (wmt 2016 shared task). In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 740–744.
- Raghavendra Udupa, K Saravanan, A Kumaran, and Jagadeesh Jagarlamudi. 2009. Mint: A method for effective and scalable mining of named entity transliterations from large comparable corpora. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 799–807. Association for Computational Linguistics.
- Michał Ziemiński, Marcin Junczys-Dowmunt, and Bruno Pouliquen. 2016. The United Nations parallel corpus v1. 0. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 3530–3534.

# Improving Context Modeling in Neural Topic Segmentation

Linzi Xing<sup>†</sup>, Brad Hackinen<sup>‡</sup>, Giuseppe Carenini<sup>†</sup>, Francesco Trebbi<sup>§</sup>

<sup>†</sup> University of British Columbia, Vancouver, Canada

<sup>‡</sup> Ivey Business School, London, Canada

<sup>§</sup> University of California Berkeley, California, USA

{lzxing, carenini}@cs.ubc.ca

bhackinen@ivey.ca

ftrebbi@berkeley.edu

## Abstract

Topic segmentation is critical in key NLP tasks and recent works favor highly effective neural supervised approaches. However, current neural solutions are arguably limited in how they model context. In this paper, we enhance a segmenter based on a hierarchical attention BiLSTM network to better model context, by adding a coherence-related auxiliary task and restricted self-attention. Our optimized segmenter<sup>1</sup> outperforms SOTA approaches when trained and tested on three datasets. We also the robustness of our proposed model in domain transfer setting by training a model on a large-scale dataset and testing it on four challenging real-world benchmarks. Furthermore, we apply our proposed strategy to two other languages (German and Chinese), and show its effectiveness in multilingual scenarios.

## 1 Introduction

Topic segmentation is a fundamental NLP task that has received considerable attention in recent years (Barrow et al., 2020; Glavas and Somasundaran, 2020; Lukasik et al., 2020). It can reveal important aspects of a document semantic structure by splitting the document into topical-coherent textual units. Taking the *Wikipedia* article in Table 1 as an example, without the section marks, a reliable topic segmenter should be able to detect the correct boundaries within the text and chunk this article into the topical-coherent units T1, T2 and T3. The results of topic segmentation can further benefit other key downstream NLP tasks such as document summarization (Mitra et al., 1997; Riedl and Biemann, 2012a; Xiao and Carenini, 2019), question answering (Oh et al., 2007; Diefenbach et al., 2018), machine reading (van Dijk, 1981;

### Preface:

Marcus is a city in Cherokee County, Iowa, United States.

### [T1] History:

S1: The first building in Marcus was erected in 1871.

S2: Marcus was incorporated on May 15, 1882.

### [T2] Geography:

S3: Marcus is located at (42.822892, -95.804894).

S4: According to the United States Census Bureau, the city has a total area of 1.54 square miles, all land.

### [T3] Demographics:

S5: As of the census of 2010, there were 1,117 people, 494 households, and 310 families residing in the city.

Table 1: A Wikipedia sample article about *City Marcus* covering three topics: T1, T2 and T3

Saha et al., 2019) and dialogue modeling (Xu et al., 2020; Zhang et al., 2020).

A wide variety of techniques have been proposed for topic segmentation. Early unsupervised models exploit word statistic overlaps (Hearst, 1997; Galley et al., 2003), Bayesian contexts (Eisenstein and Barzilay, 2008) or semantic relatedness graphs (Glavaš et al., 2016) to measure the lexical or semantic cohesion between the sentences or paragraphs and infer the segment boundaries from them. More recently, several works have framed topic segmentation as neural supervised learning, because of the remarkable success achieved by such models in most NLP tasks (Wang et al., 2016, 2017; Sehikh et al., 2017; Koshorek et al., 2018; Arnold et al., 2019). Despite minor architectural differences, most of these neural solutions adopt Recurrent Neural Network (Schuster and Paliwal, 1997) and its variants (RNNs) as their main framework. On the one hand, RNNs are appropriate because topic segmentation can be modelled as a sequence labeling task where each sentence is either the end of a segment or not. On the other hand, this choice makes these neural models limited in how to model the context. Because some sophisticated RNNs (eg.,

<sup>1</sup>Our code will be publicly available at [www.cs.ubc.ca/cs-research/lci/research-groups/natural-language-processing](http://www.cs.ubc.ca/cs-research/lci/research-groups/natural-language-processing)

LSTM, GRU) are able to preserve long-distance information (Lipton et al., 2015; Sehikh et al., 2017; Wang et al., 2018), which can largely help language models. But for topic segmentation, it is critical to supervise the model to focus more on the local context.

As illustrated in Table 1, the prediction of the segment boundary between T1 and T2 hardly depends on the content in T3. Bringing in excessive long-distance signals may cause unnecessary noise and hurt performance. Moreover, text coherence has strong relation with topic segmentation (Wang et al., 2017; Glavas and Somasundaran, 2020). For instance, in Table 1, sentence pairs from the same segment (like  $\langle S_1, S_2 \rangle$  or  $\langle S_3, S_4 \rangle$ ) are more coherent than sentence pairs across segments (like  $S_2$  and  $S_3$ ). Arguably, with a proper way of modeling the coherence between adjacent sentences, a topic segmenter can be further enhanced.

In this paper, we propose to enhance a state-of-the-art (SOTA) topic segmenter (Koshorek et al., 2018) based on hierarchical attention BiLSTM network to better model the local context of a sentence in two complementary ways. First, we add a coherence-related auxiliary task to make our model learn more informative hidden states for all the sentences in a document. More specifically, we refine the objective of our model to encourage smaller coherence for the sentences from different segments and larger coherence for the sentences from the same segment. Secondly, we enhance context modeling by utilizing restricted self-attention (Wang et al., 2018), which enables our model to pay attention to the local context and make better use of the information from the closer neighbors of each sentence (i.e., with respect to a window of explicitly fixed size  $k$ ). Our empirical results show (1) that our proposed context modeling strategy significantly improves the performance of the SOTA neural segmenter on three datasets, (2) that the enhanced segmenter is more robust in domain transfer setting when applied to four challenging real-world test sets, sampled differently from the training data, (3) that our context modeling strategy is also effective for the segmenters trained on other challenging languages (eg., German and Chinese), rather than just English.

## 2 Related Work

**Topic Segmentation** Early unsupervised models exploit the lexical overlaps of sentences to

measure the lexical cohesion between sentences or paragraphs (Hearst, 1997; Galley et al., 2003; Eisenstein and Barzilay, 2008; Riedl and Biemann, 2012b). Then, by moving two sliding windows over the text, the cohesion between successive text units could be measured and a cohesion drop would signal a segment boundary. Even if these models do not require any training data, they only show limited performance in practice and are not general enough to handle the temporal change of the languages (Huang and Paul, 2019).

More recently, neural-based supervised methods have been devised for topic segmentation because of their more accurate predictions and greater efficiency. One line of research frames topic segmentation as a sequence labeling problem and builds neural models to predict segment boundaries directly. Wang et al. (2016) proposed a simple BiLSTM model to label if a sentence is a segment boundary or not. They demonstrated that along with engineered features based on cue phrases (eg., ‘first of all’, ‘second’), their model can achieve marginally better performance than early unsupervised methods. Later, Koshorek et al. (2018) proposed a hierarchical neural sequence labeling model for topic segmentation and showed its superiority compared with their selected supervised and unsupervised baselines. Around the same time, Badjatiya et al. (2018) proposed an attention-based BiLSTM model to classify whether a sentence was a segment boundary or not, by considering the context around it. The work we present in this paper can be seen as pushing this line of research even further by encouraging the model to more explicitly consider contextual coherence, as well as to prefer more information from the neighbor context through restricted self-attention.

Another rather different line of works first trains neural models for other tasks, and then uses these models’ outputs to predict boundaries. Wang et al. (2017) trained a Convolutional Neural Network (CNN) network to predict the coherence scores for text pairs. Sentences in a pair with large cohesion are supposed to belong to the same segment. However, their “learning to rank” framework asks for the pre-defined number of segments, which limits their model’s applicability in practice. Our selected framework overcomes this constraint by tuning a confidence threshold during the training stage. A sentence with the output probability above this threshold will be predicted as the end of a seg-

ment. Following a very different approach, Arnold et al. (2019) introduced a topic embedding layer into a BiLSTM model. After training their model to predict the sentence topics, the learned topic embeddings can be utilized for topic segmentation. However, one critical flaw of their method is that it requires a complicated pre-processing pipeline, which includes topic extraction and synset clustering, whose errors can propagate to the main topic segmentation task. In contrast, our proposal only requires the plain content of the training data without any complex pre-processing.

**Coherence Modeling** Early works on coherence modeling merely predict the coherence score for documents by tracking the patterns of entities’ grammatical role transition (Barzilay and Lapata, 2005, 2008). More recently, researchers started modeling the coherence for sentence pairs by their semantic similarities and used them for higher level coherence prediction or even other tasks, including topic segmentation. Wang et al. (2017) demonstrated the strong relation between text-pair coherence modeling and topic segmentation. They assumed that (1) a pair of texts from the same document should be ranked more coherent than a pair of texts from different documents; (2) a pair of texts from the same segment should be ranked more coherent than a pair of texts from different segments of a document. With these assumptions, they created a “quasi” training corpus for text-pair coherence prediction by assigning different coherence scores to the texts from the same segment, different segments but the same document, and different documents. Then they proposed the corresponding model, and further use this model to directly conduct topic segmentation. Following their second assumption, we propose a neural solution in which by injecting a coherence-related auxiliary task, topic segmentation and sentence level coherence modeling can mutually benefit each other.

### 3 Neural Topic Segmentation Model

Since RNN-based topic segmenters have shown success with high-quality training data, we adopt a state-of-the-art RNN-based topic segmenter enhanced with attention and BERT embeddings as our basic model. Then, we extend such model to make better use of the local context, something that cannot be done effectively within the RNN framework (Wang et al., 2018). In particular, we add a coherence-related auxiliary task and a restricted

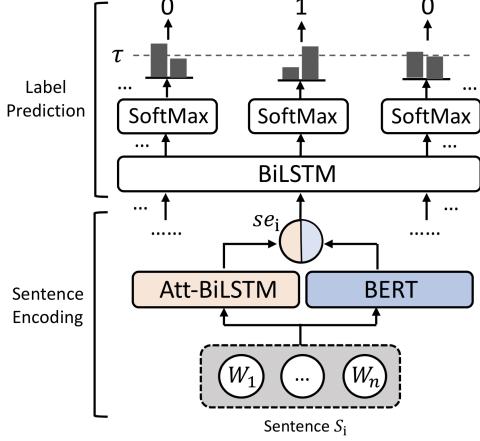


Figure 1: The architecture of our basic model.  $se_i$  is the produced sentence embedding for sentence  $S_i$ .

self-attention mechanisms to the basic model, so that predictions are more strongly influenced by the coherence between the nearby sentences. As a preview of this section, we first define the problem of topic segmentation and introduce the basic model. In the next section, we motivate and describe our proposed extensions.

#### 3.1 Problem Definition

Topic segmentation is usually framed as a sequence labeling task. More precisely, given a document represented as a sequence of sentences, our model will predict the binary label for each sentence to indicate if the sentence is the end of a topical coherent segment or not. Formally,

**Given:** A document  $d$  in the form of a sequence of sentences  $\{s_1, s_2, s_3, \dots, s_k\}$ .

**Predict:** A sequence of labels assigned to a sequence of sentences  $\{l_1, l_2, l_3, \dots, l_{k-1}\}$ , where  $l$  is a binary label, 1 means the corresponding sentence is the end of a segment, 0 means the corresponding sentence is not the end of a segment. We do not predict the label for the last sentence  $s_k$ , since it is always the end of the last segment.

#### 3.2 Basic Model: Enhanced Hierarchical Attention Bi-LSTM Network (HAN)

Figure 1 illustrates the detailed architecture of our basic model comprising the two steps of sentence encoding and label prediction. Formally, a sentence encoding network returns sentence embeddings from pre-trained word embeddings. Then a label prediction network processes the sentence embeddings generated earlier and outputs the probabilities to indicate if sentences are the segment

boundaries or not. Finally, to convert the numerical probabilities into binary labels, we follow the greedy decoding strategy in Koshorek et al. (2018) by setting a threshold  $\tau$ . All the sentences with their probabilities over  $\tau$  will be labeled 1, and 0 otherwise. This parameter  $\tau$  is set in the validation stage.

For training, we compute the cross-entropy loss between the ground truth labels  $Y = \{y_1, \dots, y_{k-1}\}$  and our predicted probabilities  $P = \{p_1, \dots, p_{k-1}\}$  for a document with  $k$  sentences:

$$L_1 = - \sum_{i=1}^{k-1} [y_i \log p_i + (1 - y_i) \log(1 - p_i)] \quad (1)$$

Looking at the details of the architecture in Figure 1, our basic model constitutes a strong baseline by extending the segmenter presented in Koshorek et al. (2018) in two ways (colored parts); namely, by improving the sentence encoder with an attention mechanism (orange) and with BERT embeddings (blue).

**Enhancing Task-Specific Sentence Representations** - While Koshorek et al. (2018) applied max-pooling to build sentence embeddings from sentence encoding network, we applied an attention mechanism (Yang et al., 2016) to make the model better capture task-wise sentence semantics. The benefit of this enhancement is verified empirically by the results in Table 2. As it can be seen, replacing the max-pooling with the attention based BiLSTM sentence encoder yields better performance.

**Enhancing Generality with BERT Embeddings**  
In order to better deal with unseen text in test data and hence improve the model’s generality, we utilize a pre-trained BERT sentence encoder<sup>2</sup> which complements our sentence encoding network. The transformer-based BERT model (Devlin et al., 2019) was trained on multi-billion sentences publicly available on the web for several generic sentence-level semantic tasks, such as Natural Language Inference and Question Answering, which implies that it can arguably capture more general aspects of sentence semantics in a reliable way. To combine task-specific information with generic semantic signals from BERT, we simply concatenate the BERT sentence embeddings with the sentence embeddings derived from our encoder. Such concatenation then becomes the input of the next level

<sup>2</sup>[github.com/hanxiao/bert-as-service](https://github.com/hanxiao/bert-as-service). For languages other than English, we use their corresponding pre-trained BERT models.

Dataset	CHOI	RULES	SECTION	MEAN
MaxPooling	1.04	7.74	12.62	7.14
BiLSTM	0.92	7.47	11.60	6.66
BERT	0.93	8.35	12.08	7.12
BiLSTM+BERT	<b>0.81</b>	<b>6.90</b>	<b>11.30</b>	<b>6.34</b>

Table 2:  $P_k$  error score (lower the better, see Section 4.3 for details) of different sentence encoding strategies on three datasets (Section 4.1). To fit in the table, we shorten Att-BiLSTM to BiLSTM. Results in **bold** are the best performance across the comparisons.

network (see Figure 1). The benefit of injecting BERT embedding is also verified empirically by the results reported in Table 2. We can see that concatenating BERT embedding and the output of Att-BiLSTM yields the best performance compared with only BERT embedding or the output of Att-BiLSTM.

### 3.3 Auxiliary Task Learning

In a well-structured document, the semantic coherence of a pair of sentences from the same segment should tend to be greater than the coherence of a pair of sentences from different segments. This observation provides us with an alternative way to enable better context modeling by formulating a coherence-related auxiliary task whose objective can be jointly optimized with our original objective (Equation 1). This task thereby is to predict the consecutive sentence-pair coherence by using the sentence hidden states generated from the BiLSTM network. Concurrently minimizing the loss of this task can regulate our model to learn better semantic coherence relation between sentences by reducing the semantic coherence scores for the sentence pairs *across segments* and increasing the semantic coherence scores for the sentence pairs *within a segment*.

To obtain the ground truth for our introduced auxiliary task (sentence-pair coherence prediction), we leverage the ground truth of our segmented training set rather than requiring external annotations. For a document which contains  $m$  sentences, there are  $m - 1$  consecutive sentence pairs. If this document has  $n$  segment boundaries, then among those  $m - 1$  sentence pairs,  $n$  sentence pairs are from different segments, while the remaining  $m - n - 1$  sentence pairs are from the same segment. In order to concurrently minimize the coherence of the sentences from different segments and maximize the coherence of the sentences in the same segment, we give a sentence pair  $\langle s_i, s_{i+1} \rangle$  a coherence label

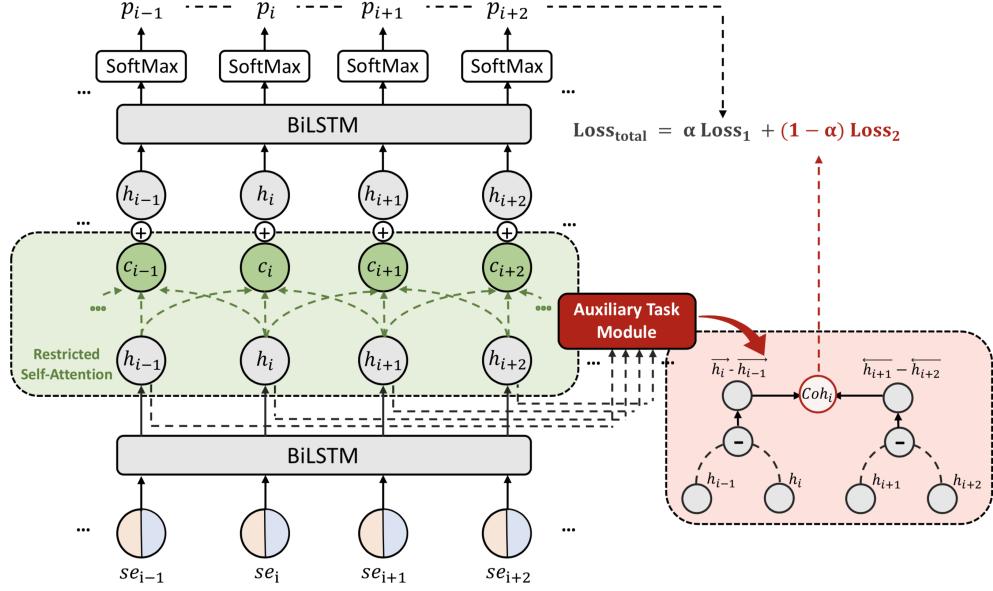


Figure 2: Our full model with context modeling components: **restricted self-attention**, **auxiliary task module**.

$l_i = 1$  if sentences in this pair are from the same segment, and  $l_i = 0$  otherwise. The embeddings  $e_i$  and  $e_{i+1}$  of adjacent sentences pairs  $\langle s_i, s_{i+1} \rangle$  used for coherence computing are calculated from BiLSTM forward and backward hidden states  $\overrightarrow{h}$  and  $\overleftarrow{h}$ , following the equations below:

$$e_i = \tanh(W_e(\overrightarrow{h_i} - \overrightarrow{h_{i-1}}) + b_e) \quad (2)$$

$$e_{i+1} = \tanh(W_e(\overleftarrow{h_{i+1}} - \overleftarrow{h_{i+2}}) + b_e) \quad (3)$$

However, notice that instead of using the conventional  $[\overrightarrow{h_i}; \overleftarrow{h_i}]$  as the embedding of sentence  $i$ , here, similarly to Wang and Chang (2016), we subtract forward/backward states to focus on the semantics of sentences in the current sentence pair. The semantic coherence between two sentence embeddings is then computed as the sigmoid of their cosine similarity:

$$Coh_i = \sigma(\cos(e_i, e_{i+1})) \quad (4)$$

We use binary cross-entropy loss to formulate the objective of our auxiliary task. For a document with  $k$  sentences, the loss can be calculated as:

$$L_2 = - \sum_{i=1, l_i=1}^{k-1} \log Coh_i - \sum_{i=1, l_i=0}^{k-1} \log(1 - Coh_i) \quad (5)$$

which penalizes high  $Coh$  across segments and low  $Coh$  within segments.

Combining Equation 1 and 5, we form the loss function of our new segmenter as:

$$L_{total} = \alpha L_1 + (1 - \alpha) L_2 \quad (6)$$

with the trade-off parameter  $\alpha$  tuned in validation stage, topic segmentation and the coherence-related auxiliary task are jointly optimized. The architecture of the auxiliary task module and its integration in our segmenter is shown in red in Figure 2.

### 3.4 Sentence-Level Restricted Self-Attention

The self-attention mechanism (Vaswani et al., 2017) has been widely applied to many sequence labeling tasks due to its superiority in modeling long-distance dependencies in text. However, when the task mainly requires modelling local context, long-distance dependencies will instead introduce noise. Wang et al. (2018) noticed this problem for discourse segmentation, where the crucial information for a clause-like Elementary Discourse Unit (EDU) boundary prediction comes usually only from the adjacent EDUs. Thus, they proposed a *word-level* restricted self-attention mechanism by adding a fixed size window constraint on the standard self-attention. In essence, this mechanism encourages the model to absorb more information directly from adjacent context words within a fixed range of neighborhood. We hypothesize that the similar restricted dependencies also play a dominant role in topic segmentation due to their close relation. Hence, instead of at word-level, we add a *sentence-level* restricted self-attention on top of the label prediction network of the basic model, as shown in green in Figure 2.

In particular, once hidden states are obtained for all the sentences of document  $d$ , we compute the

Dataset	CHOI	RULES	SECTION	WIKI-50	CITIES	ELEMENTS	CLINICAL
documents	920	4,461	21,376	50	100	118	227
# sent/seg	7.4	7.4	7.2	13.6	5.2	3.3	28.0
# seg/doc	10.0	16.6	7.9	3.5	12.2	6.8	5.0
real world	✗	✓	✓	✓	✓	✓	✓

Table 3: Statistics of all the English topic segmentation datasets used in our experiments.

Dataset	EN	DE	ZH
documents	21,376	12,993	10,000
# sent/seg	7.2	6.3	5.1
# seg/doc	7.9	7.0	6.4
real world	✓	✓	✓

Table 4: Statistics of the the WIKI-SECTION data in English(EN), German(DE) and Chinese(ZH).

similarities between the current sentence  $i$  and its nearby sentences within a window of size  $S$ . For example, the similarity between sentence  $s_i$  and  $s_j$  which is within the window size is computed as:

$$sim_{i,j} = W_a[h_i; h_j; (h_i \odot h_j)] + b_a \quad (7)$$

where  $h_i$ ,  $h_j$  are the hidden state of  $s_i$  and  $s_j$ .  $W_a$  and  $b_a$  are both attention parameters. ; is the concatenation operation and  $\odot$  is the dot product operation. The attention weights for all the sentences in the fixed window are:

$$a_{i,j} = \frac{e^{sim_{i,j}}}{\sum_{s=-S}^S e^{sim_{i,i+s}}} \quad (8)$$

The output for sentence  $i$  after the restricted self-attention mechanism is the weighted sum of all the sentence hidden states within the window:

$$c_i = \sum_{s=-S}^S a_{i,i+s} h_{i+s} \quad (9)$$

where  $c_i$  denotes the *local context embedding* of sentence  $i$  generated by restricted self-attention. After getting the local context embeddings for all the sentences, we concatenate them with the original sentence hidden states and input them to another BiLSTM layer (top of Figure 2).

## 4 Experimental Setup

In order to comprehensively evaluate the effectiveness of our context modeling strategy of adding a coherence-related auxiliary task and a restricted self-attention mechanisms to the basic model, we conduct three sets of experiments for evaluation:

(i) **Intra Domain** : we train and test the models in the same domain, repeating this evaluation for three different domains (datasets). (ii) **Domain Transfer** : we train the models on a large dataset which covers a variety of topics and test them on four challenging real-world datasets. (iii) **Multilingual** : we train and test our model on three datasets within different languages (English, German and Chinese), to assess our proposed strategy’s generality within different languages.

### 4.1 Datasets

**Data for Intra-Domain Evaluation** High quality training dataset for topic segmentation usually satisfies the following criteria: (1) large size; (2) cover a variety of topics; (3) contains real documents with reliable segmentation either from human annotations or already specified in the documents e.g., sections. In order to comprehensively evaluate the effectiveness of our context modeling strategy when dealing with data of different quality, we train and test models on the following three datasets:

**CHOI** (Choi, 2000) whose articles are synthesized artificially by stitching together different sources (i.e., they were not written as one document by one author). Hence, it does not really reflect naturally occurring topic drifts. While the quality of this dataset is low, it is an early but popular benchmark for topic segmentation evaluation. We include this dataset to allow comparison with the previous work.

**RULES** (Bertrand et al., 2018) is a dataset collected from the U.S. Federal Register issues<sup>3</sup>. When U.S. federal agencies make changes to regulations or other policies, they must publish a document called a “Rule” in the Federal Register. The Rule describes what is being changed and discusses the motivation and legal justification for the action. Since each paragraph in a document discusses one topic, we consider the last sentence of each paragraph as a ground truth topic boundary. The discussion paragraphs usually cover diverse topics in formal,

<sup>3</sup>[www.govinfo.gov/](http://www.govinfo.gov/)

technical language that can be hard to find online, so we deem it as an additional well-labelled dataset for testing topic segmentation to complement our other datasets which contain more informal use of the language.

**WIKI-SECTION** (Arnold et al., 2019) is a newly released dataset which was originally generated from the most recent English and German Wikipedia dumps. To better align with the purpose of intra-domain experiment, we only select the English samples for training and the German samples will be used in the experiments of multilingual evaluation. The English **WIKI-SECTION** (labeled **SECTION** in the tables) consists of Wikipedia articles from domain *diseases* and *cities*. We deem this dataset as the most reliable training source among the three datasets. It has the largest size and the two domains (*cities* and *diseases*) cover news-based samples and scientific-based samples respectively.

We split **CHOI** and **RULES** into 80% training, 10% validation, 10% testing. For **SECTION**, we follow Arnold et al. (2019) and split it into 70% training, 10% validation, 20% testing. Table 3 (left) contains the statistical details for these three sets.

**Data for Domain Transfer Evaluation** We pick **WIKI-SECTION** as our training set in this line of experiments, due to its largest size and variety of covered topics. Following previous work, we evaluate our model and baselines on four datasets that originate from different source distributions: **WIKI-50** (Koshorek et al., 2018) which consists of 50 samples randomly generated from the latest English Wikipedia dump, with no overlap with training and validation data. **Cities** (Chen et al., 2009) which consists of 100 samples generated from Wikipedia about cities. We also ensure that this dataset has no overlap with training and validation data. **Elements** (Chen et al., 2009) which consists of 118 samples generated from Wikipedia about chemical elements. **Clinical Books** (Malioutov and Barzilay, 2006) which consists of 227 chapters from a medical textbook. Table 3 (right) gives more detailed statistics for these datasets.

**Data For Multilingual Evaluation** In order to test the effectiveness of our context modeling strategy across languages, besides the English **WIKI-SECTION**, we train and test our model on two other Wikipedia datasets in German and Chinese: **SECTION-DE** which was released together with English **WIKI-SECTION** in Arnold et al. (2019). It

Dataset	CHOI	RULES	SECTION	MEAN
Random	49.4	50.6	51.3	50.4
BayesSeg	20.8	41.5	39.5	33.9
GraphSeg	6.6	39.3	44.9	30.3
TextSeg	1.0	7.7	12.6	7.1
Sector	-	-	12.7	-
Transformer	4.8	9.6	13.6	9.3
Basic Model	0.81	7.0	11.3	6.4
+AUX	0.64 <sup>†</sup>	6.1 <sup>†</sup>	10.4 <sup>†</sup>	5.7
+RSA	0.72 <sup>†</sup>	6.3 <sup>†</sup>	10.0 <sup>†</sup>	5.7
+AUX+RSA	<b>0.54<sup>†</sup></b>	<b>5.8<sup>†</sup></b>	<b>9.7<sup>†</sup></b>	<b>5.3</b>

Table 5:  $P_k$  error score on three datasets. Results in **bold** indicate the best performance across all comparisons. Underlined results indicate the best performance in the bottom section. <sup>†</sup> indicates the result is significantly different ( $p < 0.05$ ) from basic model.

also contains articles about cities and diseases. The section marks are used as the ground truth labels. **SECTION-ZH** which was randomly generated from the Chinese Wikipedia dump<sup>4</sup> mentioned in Hao and Paul (2020). As before, section marks are also used here as ground truth boundaries. The statistical details of these two datasets can be found in Table 4.

## 4.2 Baselines

These include two popular unsupervised topic segmentation methods, *BayesSeg* (Eisenstein and Barzilay, 2008) and *GraphSeg* (Glavaš et al., 2016), as well as the three recently proposed supervised neural models, *TextSeg* (Koshorek et al., 2018) (from which we derive our basic model), *Sector* (Arnold et al., 2019) and *Hierarchical Transformer* (labeled *Transformer* in the tables) (Glavaš and Somasundaran, 2020). We use the original implementation of *BayesSeg*, *GraphSeg* and *TextSeg*. We reimplement the *Hierarchical Transformer* ourselves. In Table 6, we adopt the results of *BayesSeg*, *GraphSeg* and *Sector* from Arnold et al. (2019)<sup>5</sup>.

## 4.3 Evaluation Metric

We use the standard  $P_k$  error score (Beeferman et al., 1999) as our evaluation metric, since it has become the standard for comparing topic segmenters.  $P_k$  is calculated as:

$$P_k(\text{ref}, \text{hyp}) = \sum_{i=0}^{n-k} \delta_{\text{ref}}(i, i+k) \neq \delta_{\text{hyp}}(i, i+k)$$

<sup>4</sup><https://linguatools.org/tools/corpora/wikipedia-monolingual-corpora/>

<sup>5</sup>Arnold et al. (2019) reported *Sector*'s performance on multiple model settings. Here we pick the performance of the model trained on *wikifull* to be close to our training setting.

Dataset	Wiki-50	Cities	Elements	Clinical
Random	52.7	47.1	50.1	44.1
BayesSeg	49.2	36.2	<b>35.6</b>	57.2
GraphSeg	63.6	40.0	49.1	64.6
TextSeg	28.5	19.8	43.9	36.6
Sector	28.6	33.4	42.8	36.9
Transformer	29.3	20.2	45.2	35.6
Basic Model	28.7	17.9	43.5	33.8
+AUX	27.9	17.0 <sup>†</sup>	41.8 <sup>†</sup>	31.5 <sup>†</sup>
+RSA	27.8 <sup>†</sup>	16.8 <sup>†</sup>	42.7	31.9 <sup>†</sup>
+AUX+RSA	<b>26.8<sup>†</sup></b>	<b>16.1<sup>†</sup></b>	39.4 <sup>†</sup>	<b>30.5<sup>†</sup></b>

Table 6:  $P_k$  error score on four test sets. Results in **bold** indicate the best performance across all comparisons. Underlined results indicate the best performance in the bottom section.  $\dagger$  indicates the result is significantly different ( $p < 0.05$ ) from basic model.

where  $\delta$  is an indicator function which is 1 if sentence  $i$  and  $i + k$  are in the same segment, 0 otherwise. It measures the probability of mismatches between the ground truth segments ( $ref$ ) and model predictions ( $hyp$ ) within a sliding window  $k$ . As a standard setting which has been used in previous work, window size  $k$  is the average segment length of  $ref$ . Since  $P_k$  is a penalty metric, lower score indicates better performance.

#### 4.4 Neural Model Setup

Following Koshorek et al. (2018), our initial word embeddings are GoogleNews word2vec ( $d = 300$ ). We also use word2vec embeddings ( $d = 300$ ) and Fasttext embeddings ( $d = 300$ ), which are both derived from Wikipedia corpora for German and Chinese respectively. We use the Adam optimizer, setting the learning rate to 0.001 and batch size to 8. The BiLSTM hidden state size is 256 following Koshorek et al. (2018). Model training is done for 10 epochs and performance is monitored over the validation set. We generate BERT sentence embeddings with the pre-trained 12-layer model released by Google AI (embedding size 768). The window size of restricted self-attention is 3 and  $\alpha$  is 0.8. These were tuned on the validation sets of the datasets we use.

### 5 Results and Discussion

#### 5.1 Intra-Domain Evaluation

Table 5 shows the models’ performance on the three datasets, when all supervised models are trained and evaluated on the training and test set from the same domain. To investigate the effectiveness of auxiliary task (AUX) and restricted self-attention

Dataset	EN	DE	ZH
Random	51.3	48.7	52.2
Basic Model	11.3	18.2	20.5
+AUX	10.4 <sup>†</sup>	17.7	20.5
+RSA	10.0 <sup>†</sup>	16.6 <sup>†</sup>	<b>19.8<sup>†</sup></b>
+AUX+RSA	<b>9.7<sup>†</sup></b>	<b>15.9<sup>†</sup></b>	20.0 <sup>†</sup>

Table 7:  $P_k$  error score on the datasets in three languages (English, German and Chinese).

(RSA), Table 5 also shows the results of individually adding each component to our basic segmenter. The most important observation from the table is that our model enhanced by context modeling outperforms all the supervised and unsupervised baselines with a substantial performance gain. With our context modeling strategy, the average  $P_k$  scores of our model over the three datasets improves on the best model (TextSeg) among the baselines by 25%. Compared with the basic model, adding AUX or RSA equally gives significant and consistent improvement across all three sets. Adding both AUX and RSA results in the biggest improvement by up to 17% on the mean across the three datasets.

#### 5.2 Domain Transfer Evaluation

Table 6 compares the performance of the baselines and our model on four challenging real-world test datasets. All supervised models are trained on the training set of *WIKI-SECTION*. One important observation is that our model enhanced by context modeling outperforms all the baseline methods on three out of four test sets with a substantial performance gap. Admittedly, *BayesSeg* performs better on *Elements*, possibly because that merely word embedding similarity is sufficient to indicate segment boundaries in this dataset. However, *BayesSeg* is completely dominated by our model on the other test sets. Overall, this indicates that our proposed context modeling strategy can not only enhance the model under the intra-domain setting, but also produce robust models that transfer to other unseen domains. Furthermore, we observe that AUX and RSA are both necessary for our model, since they do not only improve performance individually, but they achieve the best results when synergistically combined.

#### 5.3 Multilingual Evaluation

Table 7 shows results for our context modeling strategy across three different languages: English (EN), German (DE) and Chinese (ZH). Remark-

ably, even our basic model without any add-on component outperforms the random baseline by a wide margin. Looking at the gains from AUX and RSA, for German we observe a pattern similar to English, with our complete context modeling strategy (AUX+RSA) delivering the strongest gains. However, the performance on Chinese is not as strong as on English and German. Employing RSA still achieves a statistically significant  $0.7 P_k$  score drop, but introducing AUX does not help. One possible reason is that the sentences in the Chinese Wikipedia pages are relatively short and fragmented. Thus, the semantics of these sentences may be too simple to sufficiently guide the coherence auxiliary task. In general, when comparing the behavior of our context modeling strategy across these three languages, RSA appears to yield stable benefits, while the effectiveness of AUX seems to depend more on peculiarities of the dataset in the target language.

## 6 Conclusions and Future Work

We address a serious limitation of current neural topic segmenters, namely their inability to effectively model context. To this end, we propose a novel neural model that adds a coherence-related auxiliary task and restricted self-attention on top of a hierarchical BiLSTM attention segmenter to make better use of the contextual information. Experimental results of intra-domain on three datasets show that our strategy is effective within domains. Further, results on four challenging real-world benchmarks demonstrate its effectiveness in domain transfer settings. Finally, the application to other two languages (German and Chinese) suggests that our strategy has its potential in multilingual scenarios.

As future work, we will investigate whether our proposed context modeling strategy is also effective for segmenting dialogues (Takanobu et al., 2018) rather than just standard articles. Secondly, we will explore how to capture even more accurate and informative contextual information by integrating document structures or sentence dependencies obtained from other NLP tasks (e.g., discourse parsing (Huber and Carenini, 2019, 2020) or discourse role labeling (Zeng et al., 2019)).

## Acknowledgments

We thank the anonymous reviewers and the UBC-NLP group for their insightful comments.

## References

- Sebastian Arnold, Rudolf Schneider, Philippe Cudré-Mauroux, Felix A. Gers, and Alexander Löser. 2019. Sector: A neural model for coherent topic segmentation and classification. *Transactions of the Association for Computational Linguistics*, 7:169–184.
- Pinkesh Badjatiya, Litton J. Kurisinkel, Manish Gupta, and Vasudeva Varma. 2018. Attention-based neural text segmentation. In *European Conference on Information Retrieval 2018*, pages 180–193.
- Joe Barrow, Rajiv Jain, Vlad Morariu, Varun Manjunatha, Douglas Oard, and Philip Resnik. 2020. A joint model for document segmentation and segment labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 313–322.
- Regina Barzilay and Mirella Lapata. 2005. Modeling local coherence: An entity-based approach. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 141–148.
- Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34.
- Doug Beeferman, Adam Berger, and John Lafferty. 1999. Statistical models for text segmentation. *Machine Learning*, 34(1):177–210.
- Marianne Bertrand, Matilde Bombardini, Raymond Fisman, Bradley Hackinen, and Francesco Trebbi. 2018. Hall of mirrors: Corporate philanthropy and strategic advocacy. Technical report, National Bureau of Economic Research.
- Harr Chen, S.R.K. Branavan, Regina Barzilay, and David R. Karger. 2009. Global models of document structure using latent permutations. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 371–379.
- Freddy Y. Y. Choi. 2000. Advances in domain independent linear text segmentation. In *1st Meeting of the North American Chapter of the Association for Computational Linguistics*.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.
- Dennis Diefenbach, Vanessa Lopez, Kamal Singh, and Pierre Maret. 2018. Core techniques of question answering systems over knowledge bases: a survey. *Knowledge and Information Systems*, 55(3):529–569.

- Teun van Dijk. 1981. Episodes as units of discourse analysis. *Analyzing Discourse: Text and Talk*.
- Jacob Eisenstein and Regina Barzilay. 2008. Bayesian unsupervised topic segmentation. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 334–343.
- Michel Galley, Kathleen McKeown, Eric Fosler-Lussier, and Hongyan Jing. 2003. Discourse segmentation of multi-party conversation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, pages 562–569.
- Goran Glavaš, Federico Nanni, and Simone Paolo Ponzetto. 2016. Unsupervised text segmentation using semantic relatedness graphs. In *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics*, pages 125–130. Association for Computational Linguistics.
- Goran Glavas and Swapna Somasundaran. 2020. Two-level transformer and auxiliary coherence modeling for improved text segmentation. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI-20)*, pages 2306–2315.
- Shudong Hao and Michael J. Paul. 2020. An empirical study on crosslingual transfer in probabilistic topic models. *Computational Linguistics*, 46(1):95–134.
- Marti A. Hearst. 1997. Text tiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 23(1):33–64.
- Xiaolei Huang and Michael J. Paul. 2019. Neural temporality adaptation for document classification: Diachronic word embeddings and domain adaptation models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4113–4123. Association for Computational Linguistics.
- Patrick Huber and Giuseppe Carenini. 2019. Predicting discourse structure using distant supervision from sentiment. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP IJCNLP)*, pages 2306–2316.
- Patrick Huber and Giuseppe Carenini. 2020. Mega first discourse treebanks with structure and nuclearity from scalable distant sentiment supervision. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.
- Omri Koshorek, Adir Cohen, Noam Mor, Michael Rotman, and Jonathan Berant. 2018. Text segmentation as a supervised learning task. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 469–473.
- Zachary C. Lipton, John Berkowitz, and Charles Elkan. 2015. A critical review of recurrent neural networks for sequence learning. *CoRR*, abs/1506.00019.
- Michal Lukasik, Boris Dadachev, Gonçalo Simões, and Kishore Papineni. 2020. Text segmentation by cross segment attention. *CoRR*, abs/2004.14535.
- Igor Maloutov and Regina Barzilay. 2006. Minimum cut model for spoken lecture segmentation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 25–32.
- Mandar Mitra, Amit Singhal, and Chris Buckley. 1997. Automatic text summarization by paragraph extraction. In *Intelligent Scalable Text Summarization*.
- HyoJung Oh, Sung Hyon Myaeng, and Myung-Gil Jang. 2007. Semantic passage segmentation based on sentence topics for question answering. *Information Sciences*, 177(18):3696–3717.
- Martin Riedl and Chris Biemann. 2012a. How text segmentation algorithms gain from topic models. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 553–557.
- Martin Riedl and Chris Biemann. 2012b. Topictiling: A text segmentation algorithm based on lda. In *Proceedings of ACL 2012 Student Research Workshop*, pages 37–42.
- Swarnadeep Saha, Malolan Chetlur, Tejas Indulal Dhamecha, W M Gayathri K Wijayarathna, Red Mendoza, Paul Gagnon, Nabil Zary, and Shantanu Godbole. 2019. Aligning learning outcomes to learning resources: A lexico-semantic spatial approach. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 5168–5174.
- Mike Schuster and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45:2673–2681.
- Imran Seikh, Dominique Fohr, and Irina Illina. 2017. Topic segmentation in asr transcripts using bidirectional rnns for change detection. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*.
- Ryuichi Takanobu, Minlie Huang, Zhongzhou Zhao, Fenglin Li, Haiqing Chen, Xiaoyan Zhu, and Liqiang Nie. 2018. A weakly supervised method for topic segmentation and labeling in goal-oriented dialogues via reinforcement learning. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4403–4410.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30*, pages 5998–6008.

Liang Wang, Sujian Li, Yajuan Lv, and Houfeng Wang. 2017. Learning to rank semantic coherence for topic segmentation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1340–1344.

Liang Wang, Sujian Li, Xinyan Xiao, and Yajuan Lyu. 2016. Topic segmentation of web documents with automatic cue phrase identification and blstm-cnn. In *Natural Language Understanding and Intelligent Applications*, pages 177–188.

Wenhui Wang and Baobao Chang. 2016. Graph-based dependency parsing with bidirectional LSTM. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2306–2315.

Yizhong Wang, Sujian Li, and Jingfeng Yang. 2018. Toward fast and accurate neural discourse segmentation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 962–967.

Wen Xiao and Giuseppe Carenini. 2019. Extractive summarization of long documents by combining global and local context. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, pages 3009–3019.

Yi Xu, Hai Zhao, and Zhuosheng Zhang. 2020. Topic-aware multi-turn dialogue modeling. *CoRR*, abs/2009.12539.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489.

Jichuan Zeng, Jing Li, Yulan He, Cuiyun Gao, Michael R. Lyu, and Irwin King. 2019. What you say and how you say it: Joint modeling of topics and discourse in microblog conversations. *Transactions of the Association for Computational Linguistics*, 7:267–281.

Hainan Zhang, Yanyan Lan, Liang Pang, Hongshen Chen, Zhuoye Ding, and Dawei Yin. 2020. Modeling topical relevance for multi-turn dialogue generation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3737–3743. International Joint Conferences on Artificial Intelligence Organization.

# Contextualized End-to-End Neural Entity Linking

Haotian Chen

BlackRock

haotian.chen@blackrock.com

Andrej Zukov-Gregoric

BlackRock

andrej.zukovgregoric@blackrock.com

Xi (David) Li

BlackRock

david.li@blackrock.com

Sahil Wadhwa

University of Illinois at Urbana-Champaign\*

sahilw2@illinois.edu

## Abstract

We propose an entity linking (EL) model that jointly learns mention detection (MD) and entity disambiguation (ED). Our model applies task-specific heads on top of shared BERT contextualized embeddings. We achieve state-of-the-art results across a standard EL dataset using our model; we also study our model’s performance under the setting when hand-crafted entity candidate sets are not available and find that the model performs well under such a setting also.

## 1 Introduction

Entity linking (EL)<sup>1</sup>, in our context, refers to the joint task of recognizing named entity mentions in text through mention detection (MD) and linking each mention to a unique entity in a knowledge base (KB) through entity disambiguation (ED)<sup>2</sup>. For example, in the sentence “*The Times* began publication under its current name in 1788,” the span *The Times* should be detected as a named entity mention and then linked to the corresponding entity: *The Times*, a British newspaper. However, an EL model which disjointly applies MD and ED might easily mistake this mention with *The New York Times*, an American newspaper. Since our model jointly learns MD and ED from the same contextualized BERT embeddings, its final EL prediction is partially informed by both. As a result, it is able to generalize better.

Another common approach employed in previous EL research is candidate generation, where for each detected mention, a set of candidate entities is generated and the entities within it are ranked by a model to find the best match. Such sets are

built using hand-crafted rules which define which entities make it in and which do not. This risks (1) skipping out on valid entities which should be in the candidate set and (2) inflating model performance since often times candidate sets contain only one or two items. These sets are almost always used at prediction time and sometimes even during training. Our model has the option of not relying on them during prediction, and never uses them during training.

We introduce two main contributions:

- (i) We propose a new end-to-end differentiable neural EL model that jointly performs MD and ED and achieves state-of-the-art performance.
- (ii) We study the performance of our model when candidate sets are removed to see whether EL can perform well without them.

## 2 Related Work

Neural-network based models have recently achieved strong results across standard EL datasets. Research has focused on learning better entity representations and extracting better local and global features through novel model architectures.

**Entity representation.** Good KB entity representations are a key component of most ED and EL models. Representation learning has been addressed by Yamada et al. (2016), Ganea and Hofmann (2017), Cao et al. (2017) and Yamada et al. (2017). Sil et al. (2018) and Cao et al. (2018) extend it to the cross-lingual setting. More recently, Yamada and Shindo (2019) have suggested learning entity representations using BERT which achieves state-of-the-art results in ED.

**Entity Disambiguation (ED).** The ED task assumes already-labelled mention spans which are then disambiguated. Recent work on ED has focused on extracting global features (Ratinov et al.,

\*Work done while at BlackRock.

<sup>1</sup>Also known as A2KB task in GERBIL evaluation platform (Röder et al., 2018) and end-to-end entity linking in some literature

<sup>2</sup>Also known as D2KB task in GERBIL

2011; Globerson et al., 2016; Ganea and Hofmann, 2017; Le and Titov, 2018), extending the scope of ED to more non-standard datasets (Eshel et al., 2017), and positing the problem in new ways such as building separate classifiers for KB entities (Barrena et al., 2018).

**Entity Linking (EL).** Early work by Sil and Yates (2013), Luo et al. (2015) and Nguyen et al. (2016) introduced models that jointly learn NER and ED using engineered features. More recently, Kolitas et al. (2018) propose a neural model that first generates all combinations of spans as potential mentions and then learns similarity scores over their entity candidates. MD is handled implicitly by only considering mention spans which have non-empty candidate entity sets. Martins et al. (2019) propose training a multi-task NER and ED objective using a Stack-LSTM (Dyer et al., 2015). Finally, Poerner et al. (2019) and Broscheit (2019) both propose end-to-end EL models based on BERT. Poerner et al. (2019) model the similarity between entity embeddings and contextualized word embeddings by mapping the former onto the latter whereas Broscheit (2019) in essence do the opposite. Our work is different in three important ways: our training objective is different in that we explicitly model MD; we analyze the performance of our model when candidate sets are expanded to include the entire universe of entity embeddings; and we outperform both models by a wide margin.

### 3 Model Description

Given a document containing a sequence of  $n$  tokens  $\mathbf{w} = \{w_1, \dots, w_n\}$  with mention label indicators<sup>3</sup>  $\mathbf{y}_{md} = \{I, O, B\}^n$  and entity IDs  $\mathbf{y}_{ed} = \{j \in \mathbb{Z} : j \in [1, k]\}^n$  which index a pre-trained entity embedding matrix  $\mathbf{E} \in \mathbb{R}^{k \times d}$  of entity universe size  $k$  and entity embedding dimension  $d$ , the model is trained to tag each token with its correct mention indicator and link each mention with its correct entity ID.

#### 3.1 Text Encoder

The text input to our model is encoded by BERT (Devlin et al., 2019). We initialize the pre-trained weights from BERT-BASE.<sup>4</sup> The text input is tokenized by the cased WordPiece (Johnson et al.,

<sup>3</sup>We use standard *inside-outside-beginning* (IOB) tagging format introduced by (Ramshaw and Marcus, 1995)

<sup>4</sup><https://github.com/google-research/bert>

2017) sub-word tokenizer. The text encoder outputs  $n$  contextualized WordPiece embeddings  $\mathbf{h}$  which are grouped to form the embedding matrix  $\mathbf{H} \in \mathbb{R}^{n \times m}$ , where  $m$  is the embedding dimension. In the case of BERT-BASE,  $m$  is equal to 768.

The transformation from word level to WordPiece sub-word level labels is handled similarly to the BERT NER task, where the head WordPiece token represents the entire word, disregarding tail tokens.

BERT comes in two settings: feature-based and fine-tuned. Under the feature-based setting, BERT parameters are not trainable in the domain task (EL), whereas the fine-tuned setting allows BERT parameters to adapt to the domain task.

#### 3.2 EL model

**MD** is modeled as a sequence labelling task. Contextualized embeddings  $\mathbf{h}$  are passed through a feed-forward neural network and then softmaxed for classification over IOB:

$$\mathbf{m}_{md} = \mathbf{W}_{md}\mathbf{h} + \mathbf{b}_{md} \quad (1)$$

$$\mathbf{p}_{md} = \text{softmax}(\mathbf{m}_{md}) \quad (2)$$

where  $\mathbf{b}_{md} \in \mathbb{R}^3$  is the bias term,  $\mathbf{W}_{md} \in \mathbb{R}^{3 \times m}$  is a weight matrix, and  $\mathbf{p}_{md} \in \mathbb{R}^3$  is the predicted distribution across the  $\{I, O, B\}$  tag set. The predicted tag is then simply:

$$\hat{\mathbf{y}}_{md} = \arg \max_i \{\mathbf{p}_{md}(i)\} \quad (3)$$

**ED** is modeled by finding the entity (during inference this can be from either the entire entity universe or some candidate set) closest to the predicted entity embedding. We do this by applying an additional ED-specific feed-forward neural network to  $\mathbf{h}$ :

$$\begin{aligned} \mathbf{m}_{ed} &= \tanh(\mathbf{W}_{ed}\mathbf{h} + \mathbf{b}_{ed}) \\ \mathbf{p}_{ed} &= s(\mathbf{m}_{ed}, \mathbf{E}) \\ \hat{\mathbf{y}}_{ed} &= \arg \max_j \{\mathbf{p}_{ed}(j)\} \end{aligned} \quad (4)$$

where  $\mathbf{b}_{ed} \in \mathbb{R}^d$  is the bias term,  $\mathbf{W}_{ed} \in \mathbb{R}^{d \times m}$  is a weight matrix, and  $\mathbf{m}_{ed} \in \mathbb{R}^d$  is the same size as the entity embedding and  $s$  is any similarity measure which relates  $\mathbf{m}_{ed}$  to every entity embedding in  $\mathbf{E}$ . In our case, we use cosine similarity. Our

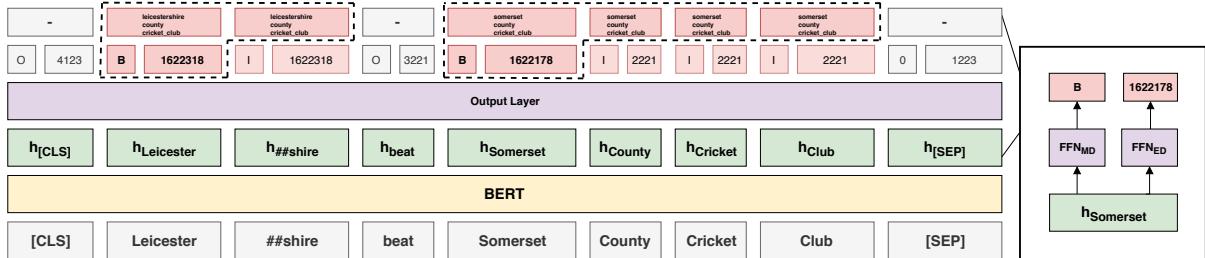


Figure 1: Architecture of the proposed model. WordPiece tokens are passed through BERT forming contextualized embeddings. Each contextualized embedding is passed through two task-specific feed-forward neural networks for MD and ED, respectively. Entity ID prediction on the ‘B’ MD tag is extended to the entire mention span.

predicted entity is the index of  $\mathbf{p}_{ed}$  with the highest similarity score.

We use pre-trained entity embeddings from *wikipedia2vec* (Yamada et al., 2018), as pre-training optimal entity representation is beyond the scope of this work. Ideally, pre-trained entity embeddings should be from a similar architecture to our EL model, but experiments show strong results even if they are not. The *wikipedia2vec* entity embeddings used in our model are trained on the 2018 Wikipedia with 100 dimensions and link graph support.<sup>5</sup>

During inference, after receiving results for each token from both the MD and ED tasks, the mention spans are tagged with  $\{B, I\}$  tags as shown in Figure 1. For each mention span, the entity ID prediction of first token represents the entire mention span. The remaining non-mention and non-first entity ID prediction are masked out. Such behavior is facilitated by the training objective below.

During training, we minimize the following multi-task objective which is inspired by Redmon and Farhadi (2017) from the object detection domain:<sup>6</sup>

$$J(\theta) = \lambda \mathcal{L}_{md}(\theta) + (1 - \lambda) \mathcal{L}_{ed}(\theta) \quad (5)$$

where  $\mathcal{L}_{md}$  is the cross entropy between predicted and actual distributions of IOB and  $\mathcal{L}_{ed}$  is the cosine similarity between projected entity embeddings and actual entity embeddings. We tentatively explored triplet loss and contrastive loss with some simple negative mining strategies for ED but did not observe significant gains in performance. The two loss functions are weighted by

<sup>5</sup><https://wikipedia2vec.github.io/wikipedia2vec/pretrained/>

<sup>6</sup>Similar to EL, object detection has two sub-tasks: locating bounding boxes and identifying objects in each box.

a hyperparameter  $\lambda$  (in our case  $\lambda = 0.1$ ). Note that  $\mathcal{L}_{md}$  is calculated for all non-pad head WordPiece tokens but  $\mathcal{L}_{ed}$  is calculated only for the first WordPiece token of every labeled entity mention with a linkable and valid entity ID label.

## 4 Experiments

### 4.1 Dataset and Performance Metrics

We train and evaluate our model on the widely used AIDA/CoNLL dataset (Hoffart et al., 2011). It is a collection of news articles from Thomson Reuters, which is split into training, validation (testa) and test (testb) sets. Following convention, the evaluation metric is strong-matching span-level InKB micro and macro F1 score over gold mentions, where entity annotation is available (Röder et al., 2018). Note that ED models are evaluated by accuracy metric while EL models are evaluated by F1, which penalizes the tagging of non-mention spans as entity mentions.

### 4.2 Candidate Sets

All EL models cited rely on candidate sets. As for our model, mentions can be efficiently disambiguated with respect to the entire entity universe, which we take to be the one million most frequent entities in 2018 Wikipedia. Consequently, our model can circumvent candidate generation, as well as the external knowledge that comes with it. In order to study the impact of candidate sets on our model, we apply candidate sets from Hoffart et al. (2011) backed by the YAGO knowledge graph (Suchanek et al., 2007). Importantly, we do not arbitrarily limit the size of the candidate sets.

### 4.3 Training Details and Settings

We train the EL model on the training split with a batch size of 4 for 50,000 steps. As in the original BERT paper, the model is optimized by the Adam

optimizer (Kingma and Ba, 2014) with the same hyperparameters except the learning rate, which we set to be 2e-5. Training was performed on a Tesla V100 GPU. A 0.1 dropout rate was used on the prediction heads. Experiments are repeated three times to calculate an error range.

## 4.4 Results

**Comparison with Other EL Models.** We compare our model with six of the most recent, and best performing, EL models in Table 1. We study the performance of our model with, and without candidate sets (see Section 4.2). We find that when candidate sets are provided, our model outperforms existing models by a significant margin.

One of the problems of comparing results in the EL and ED space is that candidate sets are usually paper-specific and many works suggest their own methodologies for generating them. In addition to using candidate sets from Hoffart et al. (2011) (which makes us comparable to Kolitsas et al. (2018) who use the same sets), we impose no arbitrary limit on candidate set size. This means that many of our candidate sets have more than the standard 20-30 candidates, which are normally considered in past works.

Without candidate sets our model also shows good results and validation performance is on par with recent work by Martins et al. (2019) who used stack LSTMs with candidate sets. We improve upon work by Broscheit (2019) who, like us, do not use candidate sets. We use a larger overall entity universe (1M instead of 700K). Interestingly, Broscheit (2019) note that during their error analysis only 3% of wrong predictions were due to erroneous span detection. This could potentially explain our margin of improvement in the test set since our model is span-aware unlike theirs. For more details on the properties of the AIDA dataset we recommend Ilievski et al. (2018).

**Overfitting.** There are considerable drops in performance between validation and test both when BERT is fine-tuned or fixed, pointing to potential problems with overfitting. Identical behaviour is seen in Broscheit (2019) and Poerner et al. (2019), who propose similar BERT-based models. Whether overfitting is due to BERT or the downstream models requires further research.

Even more considerable drops in performance between validation and test are experienced when candidates sets are not used and entities are linked

	AIDA/testa F1 (val)		AIDA/testb F1 (test)	
	Macro	Micro	Macro	Micro
Martins et al. (2019)	82.8	85.2	81.2	81.9
Kolitsas et al. (2018)	86.6	89.4	82.6	82.4
Cao et al. (2018)	77.0	79.0	80.0	80.0
Nguyen et al. (2016)	-	-	-	78.7
Broscheit (2019)	-	76.5	-	67.8
Poerner et al. (2019)	89.1	90.8	84.2	85.0
Fine-tuned BERT with candidate sets	$92.6 \pm 0.2$	$93.6 \pm 0.2$	$87.5 \pm 0.3$	$87.7 \pm 0.3$
Fine-tuned BERT without candidate sets	$82.6 \pm 0.2$	$83.5 \pm 0.2$	$70.7 \pm 0.3$	$69.4 \pm 0.3$

Table 1: Strong-matching span-level InKB macro & micro F1 results on validation and test splits of AIDA/CoNLL dataset. Note that the other models cited all use candidate sets. We run our models three times with different seeds to get bounds around our results.

Ablation	Validation F1		Test F1	
	Macro	Micro	Macro	Micro
Feature-based BERT with candidate sets	$87.1 \pm 0.1$	$90.3 \pm 0.1$	$83.5 \pm 0.3$	$84.8 \pm 0.4$
Feature-based BERT without candidate sets	$63.3 \pm 1.1$	$64.1 \pm 0.2$	$57.2 \pm 0.2$	$54.1 \pm 0.3$
With fasttext entity embedding	90.4	91.4	82.8	82.9

Table 2: Ablation results on validation and test sets of AIDA/CoNLL. By feature-based BERT we mean BERT which is not fine-tuned to the task.

across the entire entity universe. We cannot be sure whether these drops are specific to BERT since no non-BERT works cite results over the entire entity universe.

**Ablation Study.** We perform a simple ablation study, the results of which are shown in Table 2. We note that performance suffers in the EL task when BERT is not fine-tuned but still maintains strong results comparable to the state-of-the-art. Without fine-tuning, validation set performance decreases and becomes more comparable to test set performance, indicating that the fine-tuned BERT overfits in such a setting - we find this to be an interesting future direction of study.

**Other Results.** Finally, during research, we swapped the Wikipedia2Vec entities with averaged-out 300-dimensional FastText embeddings (Bojanowski et al., 2017) to see what the impact of not having entity-specific embeddings would be. To our surprise, the model performs on par with existing results which, we think, points to a combination of (1) BERT already having internal knowledge of entity-mentions given their context; and (2) many AIDA mentions being easily linkable by simply considering their surface-form. We think this too is an interesting direction of future study. Point (2) specifically points to the need for better EL datasets than AIDA, which was originally meant to be an ED dataset. A great study of point (1) can be found in Poerner et al. (2019).

## 5 Conclusions and Future Work

We propose an EL model that jointly learns the MD and ED task, achieving state-of-the-art results. We also show that training and inference without candidate sets is possible. We think that interesting future directions of study include a better understanding of how BERT already comprehends entities in text without reference to external entity embeddings. Finally, we think that moving forward, reducing the EL community’s dependence on candidate sets could be a good thing and requires more research. Dropping candidate sets could make models more easily comparable.

## References

- Ander Barrena, Aitor Soroa, and Eneko Agirre. 2018. Learning text representations for 500K classification tasks on named entity disambiguation. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 171–180, Brussels, Belgium. Association for Computational Linguistics.
- Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.
- Samuel Broscheit. 2019. Investigating entity knowledge in BERT with simple neural end-to-end entity linking. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 677–685, Hong Kong, China. Association for Computational Linguistics.
- Yixin Cao, Lei Hou, Juanzi Li, Zhiyuan Liu, Chengjiang Li, Xu Chen, and Tiansi Dong. 2018. Joint representation learning of cross-lingual words and entities via attentive distant supervision. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 227–237, Brussels, Belgium. Association for Computational Linguistics.
- Yixin Cao, Lifu Huang, Heng Ji, Xu Chen, and Juanzi Li. 2017. Bridge text and knowledge by learning multi-prototype entity mention embedding. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1623–1633, Vancouver, Canada. Association for Computational Linguistics.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. 2015. Transition-based dependency parsing with stack long short-term memory. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 334–343, Beijing, China. Association for Computational Linguistics.
- Yotam Eshel, Noam Cohen, Kira Radinsky, Shaul Markovitch, Ikuya Yamada, and Omer Levy. 2017. Named entity disambiguation for noisy text. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 58–68, Vancouver, Canada. Association for Computational Linguistics.
- Octavian-Eugen Ganea and Thomas Hofmann. 2017. Deep joint entity disambiguation with local neural attention. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2619–2629, Copenhagen, Denmark. Association for Computational Linguistics.
- Amir Globerson, Nevena Lazic, Soumen Chakrabarti, Amarnag Subramanya, Michael Ringgaard, and Fernando Pereira. 2016. Collective entity resolution with multi-focal attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 621–631.
- Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust disambiguation of named entities in text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 782–792. Association for Computational Linguistics.
- Filip Ilievski, Piek Vossen, and Stefan Schlobach. 2018. Systematic study of long tail phenomena in entity linking. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 664–674, Santa Fe, New Mexico, USA. Association for Computational Linguistics.
- Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google’s multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.
- Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*.

- Nikolaos Kolitsas, Octavian-Eugen Ganea, and Thomas Hofmann. 2018. [End-to-end neural entity linking](#). In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 519–529, Brussels, Belgium. Association for Computational Linguistics.
- Phong Le and Ivan Titov. 2018. [Improving entity linking by modeling latent relations between mentions](#). In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1595–1604, Melbourne, Australia. Association for Computational Linguistics.
- Gang Luo, Xiaojiang Huang, Chin-Yew Lin, and Zaiqing Nie. 2015. Joint entity recognition and disambiguation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 879–888.
- Pedro Henrique Martins, Zita Marinho, and André F. T. Martins. 2019. [Joint learning of named entity recognition and entity linking](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 190–196, Florence, Italy. Association for Computational Linguistics.
- Dat Ba Nguyen, Martin Theobald, and Gerhard Weikum. 2016. J-nerd: joint named entity recognition and disambiguation with rich linguistic features. *Transactions of the Association for Computational Linguistics*, 4:215–229.
- Nina Poerner, Ulli Waltinger, and Hinrich Schütze. 2019. Bert is not a knowledge base (yet): Factual knowledge vs. name-based reasoning in unsupervised qa. *arXiv preprint arXiv:1911.03681*.
- Lance Ramshaw and Mitch Marcus. 1995. [Text chunking using transformation-based learning](#). In *Third Workshop on Very Large Corpora*.
- Lev Ratnov, Dan Roth, Doug Downey, and Mike Anderson. 2011. Local and global algorithms for disambiguation to wikipedia. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1375–1384. Association for Computational Linguistics.
- J. Redmon and A. Farhadi. 2017. [Yolo9000: Better, faster, stronger](#). In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6517–6525.
- Michael Röder, Ricardo Usbeck, and Axel-Cyrille Ngonga Ngomo. 2018. [GERBIL - benchmarking named entity recognition and linking consistently](#). *Semantic Web*, 9(5):605–625.
- Avirup Sil, Gourab Kundu, Radu Florian, and Wael Hamza. 2018. Neural cross-lingual entity linking. In *Thirty-Second AAAI Conference on Artificial Intelligence*.
- Avirup Sil and Alexander Yates. 2013. Re-ranking for joint named-entity recognition and linking. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 2369–2374. ACM.
- Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: A Core of Semantic Knowledge. In *16th International Conference on the World Wide Web*, pages 697–706.
- Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2018. Wikipedia2vec: An optimized tool for learning embeddings of words and entities from wikipedia. *arXiv preprint 1812.06280*.
- Ikuya Yamada and Hiroyuki Shindo. 2019. Pre-training of deep contextualized embeddings of words and entities for named entity disambiguation. *arXiv preprint arXiv:1909.00426*.
- Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2016. [Joint learning of the embedding of words and entities for named entity disambiguation](#). In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 250–259, Berlin, Germany. Association for Computational Linguistics.
- Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2017. Learning distributed representations of texts and entities from knowledge base. *Transactions of the Association for Computational Linguistics*, 5:397–411.

# DAPPER: Learning Domain-Adapted Persona Representation Using Pretrained BERT and External Memory

Prashanth Vijayaraghavan

MIT Media lab

pralav@mit.edu

Eric Chu

MIT Media lab

echu@mit.edu

Deb Roy

MIT Media lab

dkroy@media.mit.edu

## Abstract

Research in building intelligent agents have emphasized the need for understanding characteristic behavior of people. In order to reflect human-like behavior, agents require the capability to comprehend the context, infer individualized persona patterns and incrementally learn from experience. In this paper, we present a model called DAPPER that can learn to embed persona from natural language and alleviate task or domain-specific data sparsity issues related to personas. To this end, we implement a text encoding strategy that leverages a pretrained language model and an external memory to produce domain-adapted persona representations. Further, we evaluate the transferability of these embeddings by simulating low-resource scenarios. Our comparative study demonstrates the capability of our method over other approaches towards learning rich transferable persona embeddings. Empirical evidence suggests that the learnt persona embeddings can be effective in downstream tasks like hate speech detection.

## 1 Introduction

With increasing human-machine hybrid technologies, the real-world interactions with AI systems are often stilted. This shortcoming can be attributed to the lack of shared common knowledge about how people will act, communicate and react under different circumstances. Several studies in the field of psychology (Goldberg, 1990; Barrick and Mount, 1993, 1991) have established the role of personas in governing how people process information, attend to and interpret life-experiences, and respond to social situations. Specifically, the relationship between personality and natural language have been widely studied (Digman and Takemoto-Chock, 1981; Pennebaker et al., 2003). For example, a narcissistic person might make frequent use of first-person expressions (I, me, myself, for

me, etc.). Therefore, endowing machines with the persona information can lead to the development of psychologically plausible intelligent systems. Though computational models of personality have generally followed prior psychological models or theories (Hjelle and Ziegler, 1992; Costa and PAUL, 1996), multiple definitions of personas have been in use depending on the nature of the domain or task at hand. There has been considerable amount of interest in the past that used NLP tools to conduct persona analysis of fictional characters in literary texts (Flekova and Gurevych, 2015; Mairesse et al., 2007). Motivated by such works, we focus on deriving persona representations that explain human social behavior categorized according to their influences on language, conversations and actions in different social contexts.

In this work, we define persona as the sum total of mental, emotional, and social characteristics of an individual (Soloff, 1985). This broad definition, while basing on theoretical foundations, allows us to learn persona embeddings from annotated text that span across multiple domains and social contexts. Often these persona-annotated domain data are either too small or not representative of all the domain aspects of persona. Therefore, we address these challenges by formulating our representation learning problem through the lens of domain adaptation. We propose a model called DAPPER<sup>1</sup> to learn a domain-adapted persona embedding that promotes positive knowledge transfer across multiple text domains: movies dialogue, forum discussion posts and personal life stories or essays. Towards this goal, we use a pretrained BERT model to extract rich semantic features from text and fine-tune them by introducing Adaptive Knowledge Transformer that serve as adaptive layers on top of the representations obtained from BERT model.

<sup>1</sup>Short for Domain Adapted Pretraining-based PErsona Representation

These adaptive layers enrich the representations with domain-related persona knowledge. We explore variants of Transformer encoder layer as our adaptive layers. In our experiments, we compare our Transformer-based DAPPER model with RNN-based techniques on data from three different text domains. Finally, we showcase the advantages of using our representations in a downstream hate speech detection task. Thus, our contributions are as follows:

- We propose a model called DAPPER that integrates pretrained language model with adaptive knowledge Transformer layers to learn better domain-adapted representation of personas.
- We evaluate our model on texts from multiple text domains: Movies dialogue (Chu et al., 2018), forum discussion posts and personal essays or life stories (Pennebaker and King, 1999). Our DAPPER model outperforms the baseline models significantly across these domains.
- We determine how our model performs in domains with limited labeled data by simulating such scenarios within our existing datasets. We show that our domain-knowledge enriched persona representations are capable of adapting to such domains. Further, they show promise in an unrelated downstream hate speech detection task.

## 2 Related Work

Considering that personality compels a tendency on a lot of aspects of human behavior, there have been several studies intended to model personality traits from text. An earlier work by (Pennebaker and King, 1999) compiled stream-of-consciousness essay dataset for an automated personality detection task. Since the Five Factor Model is widely accepted, several attempts have been made to detect personality from these essays including LIWC features or deep learning techniques (Majumder et al., 2017; Mairesse et al., 2007). (Chaudhary et al., 2013) compared different machine learning models to predict Myers-Brigg Type Indicator. Another line of work (Liu et al., 2016) related to personas focused on developing a language independent and compositional model for personality trait recognition for short tweets. Additionally, there have been

Datasets	Label Type	Size	# Categories
Personal Essays	Big-Five	2,400	5
Forum Posts	MBTI	52,648	16
Movies Dialogue	Tropes	17,342	72

Table 1: Details of the datasets from different domains

other efforts that model personas of movie characters and incorporate speaker persona in dialogue models based on speaking style characterized by natural language sentences (Bamman et al., 2013). We observe that most of these works use different theories and definitions for modeling personas – ranging from widely accepted psychological tests to simple emotion states of people as means of ascertaining personality (Shuster et al., 2018). However, there is very limited work (Li et al., 2016; Chu et al., 2018) focusing on persona embeddings that can be adapted to different domains. In this work, our goal is to produce general purpose persona embeddings computed using texts from various domains .

## 3 Datasets

Towards learning a domain-adapted persona embedding, we aggregate different forms of text data: (a) personal stories/essays, (b) dialogues and (c) discussion forum posts. Each of these datasets have distinct persona categories. Table 1 shows the details of the dataset. We elaborate them in the following sections.

### 3.1 Personal Essays Corpus

Personal stories or reflections explain important parts of one’s personality including their goals and values (McAdams and Manczak, 2015). For our purpose, we make use of personal essays originally from Pennebaker et al. (Pennebaker and King, 1999). This corpus consists of 2400 essays collected between 1997 and 2004. Students who produced these texts were assessed based on Big Five<sup>2</sup> Questionnaires. To obtain labels from the self-assessments, z-scores were computed from them by (Mairesse et al., 2007) and the resulting scores were discretized to categories by (Celli et al., 2013).

### 3.2 Forum Posts Corpus

One of the most commonly administered psychological tests is Myers-Briggs Type Indicator

<sup>2</sup>[https://en.wikipedia.org/wiki/Big\\_Five\\_personality\\_traits](https://en.wikipedia.org/wiki/Big_Five_personality_traits)

(MBTI<sup>3</sup>). Based on Jung’s theory of psychological types, 16 personality types were recognized as useful reference points to understand one’s personality. In this work, we collect a text corpus from a discussion forum called PersonalityCafe<sup>4</sup>, that has dedicated communities for each of the 16 MBTI personality types. The members of these communities generally self-identify with the corresponding personality type and post various forms of text including those written in a stream-of-consciousness style. To obtain these posts, we crawled specific sections of the forum related to each personality type. Further, we filter the posts that are too short (i.e. less than 75 characters in length) and replace explicit mentions of their personality type in the text with markers. Though the prevalence of MBTI personality types in general population is highly disproportional, the forum posts might not always reflect that distribution. Therefore, we create a more or less balanced dataset to avoid any skewed representation of personality types. In total, our Forum Posts dataset contains 52,648 posts. The dataset will be made publicly available.

### 3.3 Movies Dialogue Corpus

In a contrast to prior datasets which has well-defined persona categories based on personality tests/theories, we use a dataset that views character tropes as a proxy for persona labels. In the context of fiction, character trope refers to the aspects of a story that conveys information about a character including its role in the plot, personality, motivations and perceived behavior. Thus, we utilize the IMDB dialogue snippet dataset<sup>5</sup> (Chu et al., 2018) containing utterances of characters in movies obtained from CMU Movie Summary datasets (Bamman et al., 2013). Each of the 433 characters in the dataset is associated with one among 72 different trope labels. Additionally, we collect more persona-related domain-specific knowledge from TVTropes. TVTropes is a wiki that collects document descriptions about plot conventions and devices. It also contains useful notes describing MBTI<sup>6</sup> and Big Five<sup>7</sup> personality traits with references to character tropes that closely relate to each of those categories.

<sup>3</sup>[https://en.wikipedia.org/wiki/Myers–Briggs\\_Type\\_Indicator](https://en.wikipedia.org/wiki/Myers–Briggs_Type_Indicator)

<sup>4</sup><https://www.personalitycafe.com>

<sup>5</sup><https://pralav.github.io/emnlp-personas/>

<sup>6</sup><https://tvtropes.org/pmwiki/pmwiki.php/UsefulNotes/MyersBriggs>

<sup>7</sup><https://tvtropes.org/pmwiki/pmwiki.php/UsefulNotes/BigFivePersonalityTraits>

Figure 1 displays samples from the datasets used in this work. Using these datasets and persona category knowledge, we focus on developing domain-adapted persona embeddings.

<p style="color: #0070C0; font-weight: bold;">Personal Essays Corpus: PersonalityCafe — Extrovert</p> <p>.... I have some really random thoughts. I want the best things. But I fear that I want too much! What if I fall flat on my face and don't amount to anything. But I feel like I was born to do BIG things on this earth. But who knows... There is this Persian party today. My neck hurts ....</p>
<p style="color: #0070C0; font-weight: bold;">Forum Posts Corpus: PersonalityCafe — ISFJ</p> <p>#13 • May 15, 2011</p> <p>I'm tired of people making ad hominem attacks. I'm tired of people thinking they're better than me because I'm an F. I still don't believe that Americans care as much about "immigration status" as they care about the color of your skin.</p>
<p style="color: #0070C0; font-weight: bold;">Movie Dialogue Corpus: IMDB Dialogue Snippet</p> <p>Stacks Edwards: What time is it? Tommy DeVito: It's eleven thirty, we're supposed to be there by nine. Stacks Edwards: Be ready in a minute. Tommy DeVito: Yeah, you were always fuckin' late, you were late for your own fuckin' funeral. [shoots him]</p>

Figure 1: Samples from different datasets used for learning domain-adapted persona embeddings.

## 4 Problem Setup

The overall goal of our model is to learn persona embeddings using documents from different domains  $\mathcal{D}$ : dialogue utterances, forum posts and personal essays. This persona representation learning problem is formulated as a supervised classification problem. Let us denote the  $i^{th}$  input document as  $\mathcal{I}^{(i)} = [\mathcal{I}_1^{(i)}, \mathcal{I}_2^{(i)}, \dots, \mathcal{I}_{|\mathcal{I}|}^{(i)}]$ . Here, a document refers to a list of sentences from the personal essays or forum Posts corpus and dialogue snippets in case of movies dialogue corpus (explained in Section 3). Each input  $\mathcal{I}^{(i)}$  in our data is associated with their domain-specific persona label  $p_k^{(i)}$  where  $k \in \{1, 2, \dots, |\mathcal{D}|\}$ ,  $p_k^{(i)} \in \mathcal{Y}_k$  and  $\mathcal{Y}_k$  is the personal categories related to the  $k^{th}$ -domain.

## 5 Proposed Model

In this work, we explore the idea of leveraging a pretrained BERT model towards our goal of learning domain-adaptive persona embeddings. Instead of relying only on the domain-specific training data, we allow additional domain knowledge to be injected into our model using an external memory. Our model architecture is illustrated in Figure 2.

### 5.1 Input Processing

The input to our DAPPER model can take different forms depending on the domain under considera-

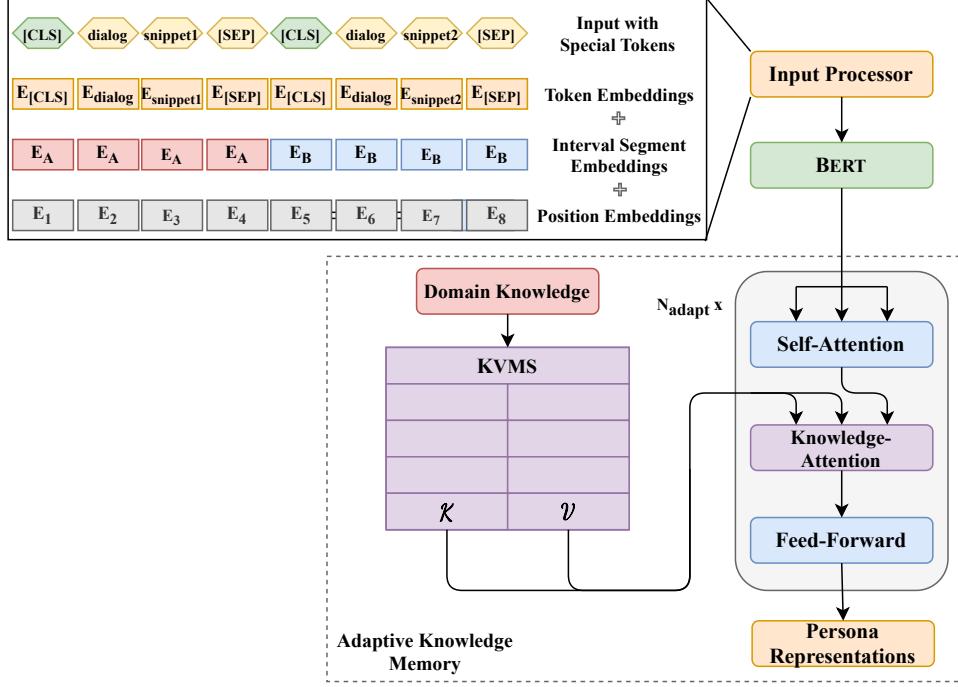


Figure 2: Illustration of our DAPPER model.

tion: (a) long essays or forum posts containing several sentences representing personal details, goals and values, and (b) dialogue snippets having character's own lines and additional contextual information such as narrator or interacting characters' lines. The varying nature of the data from these domains can pose a challenge to our modeling objective. In order to represent data from these domains, we define the following procedure:

- For Personal Essays and Forum Posts Corpus, we insert a special  $[CLS]$  token at the beginning of each sentence  $s_j$  in an essay or post with an intention that each  $[CLS]$  token will accumulate the features of the tokens following it.
- For Dialogue Corpus, we introduce a  $[CLS]$  before every dialogue snippet  $d_j$  while the character's own lines and additional context are separated by a  $[SEP]$  token.
- Next, we apply interval segment embeddings,  $E_A$  or  $E_B$ , to distinguish sentences or dialogue snippets in our data. This is done by alternating assignments between two consecutive sentences or dialogue snippets. For example, we would assign  $[E_A, E_B, E_A, E_B]$  to a list of dialogue snippets denoted as  $[d_1, d_2, d_3, d_4]$ .

We also incorporate position embeddings into our

input data processing step. Thus, we obtain a uniform way of representing our inputs texts from different domains. This allows us to hierarchically learn abstract persona representations.

## 5.2 Encoder

Our input document  $\mathcal{I}^{(i)}$  is passed to our input processing module  $f_{\mathcal{I}}(\cdot)$ . The output of this module is a document representation augmented with special tokens and processed with interval segment and position embeddings. The processed input is passed to the pretrained BERT model. Formally, this is computed as:

$$H^{(i)} = \text{BERT}(f_{\mathcal{I}}(\mathcal{I}^{(i)})) \quad (1)$$

where  $f_{\mathcal{I}}$  is the input processing function,  $H^{(i)}$  contains contextualized embeddings related to each token in the processed input document. We obtain  $j^{th}$  sentence or snippet embeddings by extracting the corresponding vector of  $j^{th}$   $[CLS]$  token from the topmost BERT layer. We denote this as  $R^{(i)} \in \mathcal{R}^{|I| \times d_h}$ ,  $d_h$  is the set to the hidden dimensions of the BERT model.

## 5.3 Adaptive Knowledge Transformer

Inspired by a prior work by (Miller et al., 2016; Zhang et al., 2017), we integrate an external memory module with the Transformer architecture and refer it as Adaptive Knowledge Transformer (AKT).

This component aids to create persistent latent embeddings related to persona categories and further accumulate more knowledge as we process data from new domains. We conceptualize this component to be composed of: (a) a Key-Value Memory Store (KVMS) that specifically facilitates adaptivity to new domains or data (b) Transformer-based adaptive layers that attends over the contents of the memory to enrich the representation with persona-related domain knowledge. By feeding the computed  $R^{(i)}$  into our AKT, we obtain domain-knowledge enriched persona embeddings. This is given as:

$$\mathcal{P}^{(i)} = \text{AKT}(R^{(i)}) \quad (2)$$

### 5.3.1 KVMS: Key-Value Memory Store

Our KVMS module consists of a mutable key matrix ( $\mathcal{K} \in \mathcal{R}^{N_M \times d_K}$ ) that accumulates persona-related knowledge across multiple domains and a non-updatable value matrix ( $\mathcal{V} \in \mathcal{R}^{N_M \times d_V}$ ) containing a learnable persona category embedding. The key matrix,  $\mathcal{K}$ , is initialized with representations of text descriptions of character tropes, MBTI types and Big-Five traits collected from TVTropes wiki (explained in 3.3) while the value matrix,  $\mathcal{V}$ , is set to their corresponding learnable persona category embeddings. We feed the text descriptions through the input processing model and compute the sum of the sentence embeddings obtained from the topmost layer of BERT.

### 5.3.2 Knowledge-Attention

Conventionally, a Transformer encoder layer consists of two sub-layers: (a) a multi-headed self-attention network and (b) a point-wise fully-connected network. Each sub-layer has a residual connection followed by layer normalization. For the sake of brevity, we avoid the residual connections and layer normalization functions in our model illustration (Figure 2) and explanation.

Our Transformer-based adaptive layers contain an additional sub-layer to integrate the persona-relevant domain knowledge into the contextual representation obtained from the encoder. We refer to this sub-layer as Knowledge-Attention. This is fine-tuned using domain-specific categories based on a supervised classification objective. The steps involved in Transformer adaptive layers are given

as follows:

$$Q^{(n)} = \text{MHA}(C^{(n-1)}, C^{(n-1)}, C^{(n-1)}) \quad (3)$$

$$A^{(n)} = \text{MHA}(Q^{(n)}, \mathcal{K}, \mathcal{V}) \quad (4)$$

$$C^{(n)} = \text{FFN}(A^{(n)}) \quad (5)$$

$$\mathcal{P}^{(i)} = C^{N_{adapt}} \quad (6)$$

where MHA is a multi-head attention function as explained in (Vaswani et al., 2017),  $n = \{1, 2, \dots, N_{adapt}\}$ ,  $C^{(0)} = R^{(i)}$ ,  $C^{(n-1)}$  is the output from the previous Transformer layer,  $A^{(n)}$  is the output from the knowledge-attention sub-layer. Our knowledge-attention mechanism identifies the most correlated and relevant knowledge from the KVMS component with respect to the input document embeddings. The resulting domain knowledge-enhanced representations are fed to the point-wise feed-forward sub-layer (FFN). We stack such adaptive layers on top of each other and the output from  $N_{adapt}^{th}$  layer is our final domain-adapted persona representation,  $\mathcal{P}^{(i)}$ .

### 5.3.3 Memory Update

Intuitively, accumulation of persona-related knowledge extracted from the training documents into our memory store can enhance the quality of the learned persona embeddings. Therefore, we perform a memory update operation on selective rows in the key matrix  $\mathcal{K}$  based on the persona-related features derived from the input document and its corresponding ground truth persona labels. The update step is defined as follows:

$$\lambda = \sigma(W_k \mathcal{K}[g_j] + W_r \phi(\mathcal{P}^{(i)})) \quad (7)$$

$$\mathcal{K}[g_j] = \lambda \odot \mathcal{K}[g_j] + (1 - \lambda) \odot \phi(\mathcal{P}^{(i)}) \quad (8)$$

where  $g_j$  refers to the indices of the rows in KVMS containing knowledge about ground truth persona label  $p_k^{(i)}$ ,  $\phi$  is aggregation function that compresses the information from  $\mathcal{P}^{(i)}$  into a single vector. We find from preliminary experiments that the mean [CLS] token embedding serves as an effective alternative to computing an average embedding related to the tokens in the input document.

## 5.4 Training Objective

Our model learns persona embeddings using a supervised classification objective. We feed the output of the aggregation function  $\phi$  to a domain-specific softmax layer to get  $q$ , where  $q =$

$\text{softmax}(f_q(\phi(\mathcal{P}^{(i)})))$ . Note that the categories vary across each domain.

$$\mathcal{L}_{CE} = \sum_{j=1}^{N_k} -p_j \log(q_j) \quad (9)$$

$$\mathcal{L}_{attn} = \frac{1}{M} \sum_{j=1}^M -\log(r_j[g_j]) \quad (10)$$

$$\mathcal{L} = \alpha_1 \mathcal{L}_{CE} + \alpha_2 \mathcal{L}_{attn} \quad (11)$$

where  $\mathcal{L}_{CE}$  is the cross-entropy loss,  $\alpha_1, \alpha_2$  are learnable parameters,  $p_j \in \{0, 1\}$  denotes the ground-truth label that reflects if the input document belongs to  $j^{th}$  persona category,  $\mathcal{L}_{attn}$  is the attention loss that promotes focus on rows with ground truth persona,  $r_j[g_j]$  is the attention score for the row in  $\mathcal{K}$  reflecting  $p_k^i$ 's knowledge.

## 6 Experiments

In this section, we describe the various evaluations settings: datasets, baselines, our model variants, modes and metrics. Our experiments are designed to study the following research questions:

**RQ1:** How well does our DAPPER model perform in comparison to baselines and its variants on domain-specific persona classification task?

**RQ2:** Is our model capable of adapting to new domains with limited labeled data?

**RQ3:** How good are the learned persona embeddings? Do they exhibit transfer capability to a downstream task?

### 6.1 Dataset Preparation

We evaluate our models using persona-related datasets from different domains: movies dialogue, forum posts and personal essays as explained in Section 3. Using a 70-10-20 split, we divide our persona dataset associated with each domain into training, validation and test sets.

### 6.2 Baselines & Model Variants (RQ1)

Since we collect persona datasets from different domains, we also compare our model's performance to domain-specific baseline methods. All these methods are enlisted as follows:

- AFF2VEC (Khosla et al., 2018) is a method for enriched word embeddings that are representative of affective interpretations of words.
- CNN (Kim, 2014) is a single-layer CNN where the input document is passed in entirety without any additional knowledge. For Personal

Essays corpus, we report the best results from (Majumder et al., 2017) as they use additional features to improve persona classification task.

- AMN (Chu et al., 2018) learns persona embeddings from movies dialogue using a multi-level attention mechanism augmented with prior knowledge about persona categories. Note that this model is one of the closest relevant work to our model. For movies dialogue corpus, we report scores only for the best performing configuration, i.e.,  $ndialog = 32$ . For the remaining datasets, we treat each sentence from the text as a character utterance and train the model accordingly.
- TTS is a non-pretrained Transformer baseline trained with the same settings as (Vaswani et al., 2017). We do not feed additional domain knowledge to this model. It is randomly initialized and trained for our task from the scratch.
- BERT FT (Devlin et al., 2018) is a fine-tuned (FT) version of BERT<sub>base</sub> model. We do not feed additional domain knowledge to this model. We refrain from training BERT<sub>large</sub> due to memory constraints.
- BERT + GRU FT (Devlin et al., 2018; Chung et al., 2014) is a similar to our DAPPER model, but applies GRU-based adaptive for persona classification task. For this setting, we experiment with and without additional knowledge using a suffix “+K”. In “+K” setting, we use GRU as the controller and apply an approach similar to AMN to enrich the learnt embeddings with domain knowledge. Without the suffix, GRU is used for fine-tuning only.
- DAPPER is our complete model by default. We also experiment with its variants using suffix “-K” indicating no knowledge attention.

The various BERT-based models can be considered as variants of our DAPPER model. While we report  $F_1$ -scores for movies dialogue and forum discussion post datasets, we report accuracy scores for personal essays corpus in order to remain consistent with prior work evaluation metrics (Majumder et al., 2017).

### 6.3 Model Modes (RQ2)

We attribute the domain adaptive capability of our DAPPER model to three main aspects: pretrained language model, domain knowledge enrichment and joint training across multiple datasets. However, this ability can be demonstrated only when we apply it to domains with limited labeled data. Therefore, we run our model in “ADAPT” mode which simulates low-data regimes to analyze the importance of some of the above mentioned aspects. In ADAPT mode, we restrain the amount of training data for only one of the domains while retaining the complete set for the remaining domains. Further, we vary the percentage of training examples from one domain to understand how early our models adapt to that domain (with decent performance). We refer to the default model mode for experiments in Section 6.2 as “FULL”. For this experiment, we plot the average prediction performance ( $F_1$ ) for varying percentages of domain-specific training set.

### 6.4 Other Experimental Settings

For baselines, we initialize our word embedding layers using GloVe (Pennington et al., 2014) embeddings. We use the publicly released pre-trained model parameters for BERT variants. We perform a grid-search and optimize the hyperparameters using the validation set. In our experiments,  $N_{adapt} = 3$ , resulted in best outcomes. We use Adam (Kingma and Ba, 2014) as our optimizer. In FULL mode, the model achieves the best performance after training for 50 epochs with a learning rate of  $\alpha = 0.00001$ . For ADAPT mode, we perform a fixed number of epochs to train each variant. We use PyTorch to implement our model and train it on 4 GPUs. In order to alleviate the problem of unbalanced datasets, we utilize class weights in categorical cross-entropy loss for each domain based on the training and validation sets.

## 6.5 Results

### 6.5.1 DAPPER Performance (RQ1)

Table 2 presents the results of our evaluation under complete training data settings (FULL). Our DAPPER model achieves an absolute improvement of 14.53% over previously reported model baseline (AMN) in the dialogues domain. While several models have shown only marginal improvement in prediction performance on Personal Essays corpus, our model shows promise by recording an improvement of 8.67% in comparison to the

previously reported CNN baseline. Overall, our DAPPER model outperforms the baselines across all the three datasets significantly.

**Effect of Architecture Choices (RQ1):** Pre-trained BERT-based models have consistently outperformed all the previous baselines including the non-pretrained TTS model. Moreover, the Transformer-based adaptive layers, with an average improvement of 6.1% (with knowledge-attention) and 4.2% (without knowledge-attention), are much more powerful than RNN-based adaptive layers. Further, we observe that BERT + GRU FT records only marginal gains over BERT when there is no knowledge-attention.

**Effect of Knowledge-Attention (RQ1):** From our results in Table 2, we analyze the importance of the knowledge-attention to the overall performance gain. We compute percentage performance gain between similar models with and without knowledge-attention sub-layer(eg. DAPPER, DAPPER – K). We find that the performance boost provided by the knowledge-attention module is noteworthy. We posit that the higher percentage gain (7.38%) for Forum Posts dataset is due to the additional domain knowledge (MBTI-related) ingested into our KVMS (explained in Section 3). Inspecting further within individual domain, the percentage increase in prediction performance almost doubles<sup>8</sup> for Transformer-based adaptive layers (as in DAPPER) in comparison with RNN-based adaptive layers (BERT + GRU FT + K). The reason for this phenomenon can be ascribed to the multi-hop knowledge enrichment facilitated by  $N_{adapt}$  encoder layers commonly observed in Memory networks literature (Miller et al., 2016).

### 6.5.2 ADAPT Mode Performance (RQ2)

Figure 3a and 3b show the mean prediction performance on movies dialogue and forum posts datasets respectively. We measure the domain adaptive capability of models based on the distance from its lifetime best performance. By varying the percentage of training data, we notice that our DAPPER model stabilizes early and outperforms the other variants with limited amount of training data. Notably, AMN model performs better than TTS model under low-data regimes. The improved performance of AMN is due to the domain knowledge enrichment via an external memory module.

<sup>8</sup>% increase-RNN vs Transformer-based adaptive layers:  
Movies dialogue corpus: 1.6% vs 3.24% (dialogue), 5.86% vs 8.9% (posts), 1.6% to 2.4% (essays)

Models	Domain-related Persona Datasets		
	Movies Dialogues ( $F_1$ )	Forum Posts ( $F_1$ )	Personal Essays (Acc.)
AFF2VEC	—	—	0.579*
CNN	0.628	0.391	0.588*
AMN	0.750*	0.453	0.591
TTS	0.776	0.496	0.593
BERT FT	0.804	0.539	0.607
BERT + GRU FT + K	0.820	0.579	0.616
BERT + GRU FT	0.807	0.547	0.608
DAPPER	<b>0.859</b>	<b>0.636</b>	<b>0.639</b>
DAPPER – K	0.832	0.584	0.624

(a)

Models	$F_1$
BCA	<b>0.744*</b>
CNN-CHAR	0.735*
1-Extra Feature	
BCA + $\mathcal{P}$	0.776
BCA + Sc	<b>0.784*</b>
All Features	
BCA + Sc + $\mathcal{P}^{(>att)}$	0.812
BCA + Sc + $\mathcal{P}^{(<att)}$	<b>0.824</b>

(b)

Table 2: Evaluation results of different models on: (a) three different Persona-related domain datasets in FULL mode, and (b) a downstream application – Hate Speech detection. Results with \* are taken from prior studies using the model on that dataset.

This feature is absent in TTS. Furthermore, we note that DAPPER – K model is able to maintain a good performance even under low-data settings. We intuit that pretraining involved in DAPPER – K model is one of the reasons behind this behavior. Therefore, we find that our DAPPER model is able to learn general purpose persona embeddings that can adapt to low-data settings. Moreover, the combination of pretraining and adaptive knowledge transformer facilitates domain adaptation effectively.

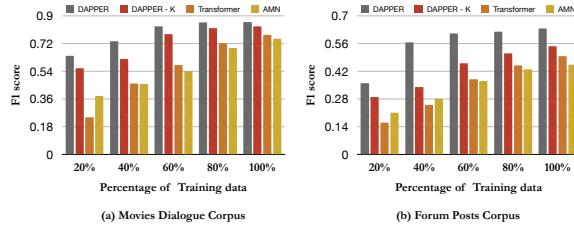


Figure 3: Evaluation of DAPPER model in ADAPT mode. We report the mean prediction performance ( $F_1$ ) on Movies Dialogue and Forum Posts dataset.

## 6.6 Cluster Analysis (RQ3)

In order to demonstrate the capabilities of our persona embedding, we first perform a simple cluster analysis. Following prior studies (Bamman et al., 2013; Chu et al., 2018), we measure the ability to recover persona-based clusters using our embeddings through the purity scores as in (Bamman et al., 2013). We compute the overlap between clusters as:  $Purity = \frac{1}{N} \sum_n \max_j |y_n \cap c_j|$ , where  $y_n$  is the  $n$ -th ground truth cluster,  $N$  is total number of characters,  $c_j$  is the  $j^{th}$  predicted cluster. By applying simple agglomerative clustering on our persona embeddings ( $k$  clusters), we report these

<b><math>k</math></b>	<b>AMN</b>	<b>DP</b>	<b>DAPPER</b>
25	48.4	39.63	<b>68.6</b>
50	48.1	31.0	<b>65.3</b>
100	45.2	24.4	<b>63.4</b>

Table 3: Cluster purity scores. DP is the Dirichlet Persona as reported in (Bamman et al., 2013)

purity scores for movies dialogue corpus. Specifically, we compare the results with AMN. Results in Table 3 indicate that our DAPPER model sharpens the persona embeddings so as to form much better clusters.

## 7 Application: Hate Speech Detection

With concerns about hate crimes, harassment, and intimidation on the rise, the role of online hate in exacerbating such violence cannot be discounted. Hence, there is an growing need to identify and counter the problem of hateful content on social media. While most prior modeling approaches have attempted to capture the semantics of hate from text, a few of them (Vijayaraghavan et al.) have used multi-modal information to detect hateful content. Few attempts have been made to study the personality of targets and instigators of hate. Since our DAPPER model learns persona embeddings from different forms of text such as dialogues, posts or personal essays, we deem it fit to explore how well our persona embeddings transfer knowledge to a hate speech detection task involving texts from a different domain (in our case, Twitter).

There are several publicly available labeled hate speech datasets (de Gibert et al., 2018; Waseem, 2016) but very few include author metadata or

tweets. In this work, we take advantage of the models and datasets introduced by (Vijayaraghavan et al.) (hereafter referred as MM-HATE). This weakly-labeled dataset contains author information and additional metadata about potential hate groups. Instead of training a powerful hate speech system from the scratch, we augment their base architecture with our persona embeddings and evaluate the prediction performance on the task at hand. We compute persona representations ( $\mathcal{P}$ ) for an author based on their past tweets. We train MM-HATE’s best performing model, BiGRU+CHAR+ATTN (BCA), under the following settings: (a) BCA +  $\mathcal{P}$ , which combines our persona embeddings with the extracted text features, (b) BCA + SC +  $\mathcal{P}^{(>att)}$ , which integrates the persona embeddings at the penultimate layer. Note that the text and socio-cultural (SC) features are already fused at that layer, and (c) BCA + SC +  $\mathcal{P}^{(<att)}$  fuses the extracted text and socio-cultural features with persona embeddings using an attention layer (as in MM-HATE).

Table 2b summarizes the results of our evaluation on hate speech detection task. We observe that SC-fused model (BCA + SC) performs marginally better than our persona-fused model (BCA +  $\mathcal{P}$ ). This result can be ascribed to the domain specificity of SC features. We also note that the combination of all the extracted features leads to a marked improvement in prediction performance, and even more so when the persona embeddings are fed to the fusion layer (BCA + SC +  $\mathcal{P}^{(<att)}$ ). Thus, our DAPPER model is able to extract behavioral features from user texts allowing positive knowledge transfer to various domains and applications.

## 8 Conclusion

We proposed a DAPPER model that learns a domain adapted pretraining-based persona representation. Our DAPPER model leverages pretrained BERT model and fine-tunes it with additional domain-adaptive layers. By introducing a knowledge-attention mechanism, we allow the domain knowledge to be integrated into our persona embeddings. The proposed model achieves significant gains across persona classification task in different domains. Our evaluations validate that our model is capable of adapting to a new domain with limited labeled data. We also highlight the transferability of our persona embeddings in a downstream hate speech detection task.

## References

- David Bamman, Brendan O’Connor, and Noah A Smith. 2013. Learning latent personas of film characters. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 352–361.
- Murray R Barrick and Michael K Mount. 1991. The big five personality dimensions and job performance: a meta-analysis. *Personnel psychology*, 44(1):1–26.
- Murray R Barrick and Michael K Mount. 1993. Autonomy as a moderator of the relationships between the big five personality dimensions and job performance. *Journal of applied Psychology*, 78(1):111.
- Fabio Celli, Fabio Pianesi, David Stillwell, and Michal Kosinski. 2013. Workshop on computational personality recognition: Shared task. In *Seventh International AAAI Conference on Weblogs and Social Media*.
- Shristi Chaudhary, Ritu Singh, Syed Tausif Hasan, and Ms Inderpreet Kaur. 2013. A comparative study of different classifiers for myers-brigg personality prediction model. *Linguistic analysis*, page 21.
- Eric Chu, Prashanth Vijayaraghavan, and Deb Roy. 2018. Learning personas from dialogue with attentive memory networks. *arXiv preprint arXiv:1810.08717*.
- Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.
- JR Costa and T PAUL. 1996. of personality theories: Theoretical contexts for the five-factor model. *The five-factor model of personality: Theoretical perspectives*, 51.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
- John M Digman and Naomi K Takemoto-Chock. 1981. Factors in the natural language of personality: Re-analysis, comparison, and interpretation of six major studies. *Multivariate behavioral research*, 16(2):149–170.
- Lucie Flekova and Iryna Gurevych. 2015. Personality profiling of fictional characters using sense-level links between lexical resources. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1805–1816.
- Ona de Gibert, Naiara Perez, Aitor García-Pablos, and Montse Cuadros. 2018. Hate speech dataset from a white supremacy forum. *arXiv preprint arXiv:1809.04444*.

- Lewis R Goldberg. 1990. An alternative” description of personality”: the big-five factor structure. *Journal of personality and social psychology*, 59(6):1216.
- Larry A Hjelle and Daniel J Ziegler. 1992. *Personality theories: Basic assumptions, research, and applications*. McGraw-Hill Book Company.
- Sopan Khosla, Niyati Chhaya, and Kushal Chawla. 2018. Aff2vec: Affect-enriched distributional word representations. *arXiv preprint arXiv:1805.07966*.
- Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.
- Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Jiwei Li, Michel Galley, Chris Brockett, Georgios P Spithourakis, Jianfeng Gao, and Bill Dolan. 2016. A persona-based neural conversation model. *arXiv preprint arXiv:1603.06155*.
- Fei Liu, Julien Perez, and Scott Nowson. 2016. A language-independent and compositional model for personality trait recognition from short texts. *arXiv preprint arXiv:1610.04345*.
- François Mairesse, Marilyn A Walker, Matthias R Mehl, and Roger K Moore. 2007. Using linguistic cues for the automatic recognition of personality in conversation and text. *Journal of artificial intelligence research*, 30:457–500.
- Navonil Majumder, Soujanya Poria, Alexander Gelbukh, and Erik Cambria. 2017. Deep learning-based document modeling for personality detection from text. *IEEE Intelligent Systems*, 32(2):74–79.
- Dan P McAdams and Erika Manczak. 2015. Personality and the life story.
- Alexander Miller, Adam Fisch, Jesse Dodge, Amir Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. *arXiv preprint arXiv:1606.03126*.
- James W Pennebaker and Laura A King. 1999. Linguistic styles: Language use as an individual difference. *Journal of personality and social psychology*, 77(6):1296.
- James W Pennebaker, Matthias R Mehl, and Kate G Niederhoffer. 2003. Psychological aspects of natural language use: Our words, our selves. *Annual review of psychology*, 54(1):547–577.
- Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.
- Kurt Shuster, Samuel Humeau, Antoine Bordes, and Jason Weston. 2018. Engaging image chat: Modeling personality in grounded dialogue. *arXiv preprint arXiv:1811.00945*.
- Paul H Soloff. 1985. Personality disorders. In *Diagnostic interviewing*, pages 131–159. Springer.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.
- Prashanth Vijayaraghavan, Hugo Larochelle, and Deb Roy. Interpretable multi-modal hate speech detection.
- Zeerak Waseem. 2016. Are you a racist or am i seeing things? annotator influence on hate speech detection on twitter. In *Proceedings of the first workshop on NLP and computational social science*, pages 138–142.
- Jiani Zhang, Xingjian Shi, Irwin King, and Dit-Yan Yeung. 2017. Dynamic key-value memory networks for knowledge tracing. In *Proceedings of the 26th international conference on World Wide Web*, pages 765–774.

# Event Coreference Resolution with Non-Local Information

Jing Lu and Vincent Ng

Human Language Technology Research Institute

University of Texas at Dallas

Richardson, TX 75083-0688

{ljwinnie, vince}@hlt.utdallas.edu

## Abstract

Existing event coreference resolvers have largely focused on exploiting the information extracted from the local contexts of the event mentions under consideration. Hypothesizing that non-local information could also be useful for event coreference resolution, we present two extensions to a state-of-the-art joint event coreference model that involve incorporating (1) a supervised topic model for improving trigger detection by providing global context, and (2) a preprocessing module that seeks to improve event coreference by discarding unlikely candidate antecedents of an event mention using discourse contexts computed based on salient entities. The resulting model yields the best results reported to date on the KBP 2017 English and Chinese datasets.

## 1 Introduction

Event coreference resolution is the task of determining the event mentions in a document that refer to the same real-world event. One of its major challenges concerns error propagation: since the event coreference resolution component typically lies towards the end of the standard information extraction pipeline, the performance of an event coreference resolver can be adversely affected by errors propagated from its upstream components. The upstream component that has the largest impact on event coreference performance is arguably *trigger detection*. Recall that the goal of a trigger detector is to identify event triggers and assign an event subtype to each of them. Failure to detect triggers could therefore limit the upper bound on event coreference performance.

To address error propagation, one way that has been shown to be effective for a variety of NLP tasks is to develop *joint* models, which allow cross-task output constraints to be learned from annotated training data. For event coreference, a learner

can easily learn, for instance, that two coreferent event mentions must have the same event subtype, thereby allowing event coreference to influence trigger detection. Unfortunately, the vast majority of existing event coreference resolvers have adopted a pipeline architecture where trigger detection precedes event coreference. In particular, joint models are both under-studied and under-exploited for event coreference given the usefulness they have demonstrated for other NLP tasks. One exception is Lu and Ng’s (2017a) joint model, which jointly learns trigger detection and event coreference and has achieved state-of-the-art results. As a structured conditional random field, the model employs unary factors to encode the features specific for each task and binary/ternary factors to capture the interaction between each pair of tasks. The use of binary/ternary factors is a particularly appealing aspect of this model: it allows these cross-task interactions to be captured in a *soft* manner, enabling the learner to *learn* which combinations of values of the output variables are more probable.

We hypothesize that the power of this joint event coreference model has not been fully exploited and seek to extend it in this paper. Our extensions are based on the observation that the strength of a joint model stems from its ability to facilitate cross-task knowledge transfer. In other words, the better we can model *each* task involved, the more we can potentially get out of joint modeling. Given this observation, we seek to improve the modeling of these tasks in this joint model as follows.

First, we improve trigger detection by exploiting topic information. State-of-the-art trigger detectors, including those based on deep neural networks (e.g., Nguyen et al. (2016)), classify each candidate trigger using local information and largely ignore the fact that the *topic* of the document in which a trigger appears plays an important role in determining its event subtype. To understand the usefulness

---

Three journalists at The New York Times on Tuesday announced plans to  $\{\text{leave}\}_{ev1}$  the newspaper. The  $\{\text{departures}\}_{ev2}$  follow moves last month by several other Times employees, all of whom were  $\{\text{leaving}\}_{ev3}$  to join digital companies. Pakistan’s Interior Ministry has ordered New York Times Reporter to  $\{\text{leave}\}_{ev4}$ . The ministry gave no explanation for the expulsion order. “You are therefore advised to  $\{\text{leave}\}_{ev5}$  the country within 72 hours,” the order stated.

---

Table 1: Event coreference resolution examples.

of document topics, consider the examples in Table 1: although all five events have similar trigger words, we can see that the meaning of the triggers and their event subtypes are different in different contexts. Hence, if an event coreference model knows that the topics of these two documents are different, it can exploit this information to more accurately classify their event subtypes. In particular, we propose to train a supervised topic model to infer the topic of each word in a test document, with the goal of understanding each candidate trigger using its *global* in addition to local context.

Second, we improve event coreference by exploiting *discourse* information. Specifically, we introduce a *preprocessing* component for event coreference resolution where we *prune* the candidate antecedents of an event mention that are unlikely to be its correct antecedent based on *discourse context*. In essence, this discourse-based preprocessing step seeks to simplify the job of the event coreference model by reducing the number of candidate antecedents it has to consider for a given event mention. We encode the discourse context of an event mention using the entities that are *salient* at the point of the discourse in which the event mention appears. To our knowledge, we are the first to show that event coreference performance can be improved using discourse contexts that are encoded using salient discourse entities.

In sum, the contributions of this paper are two-fold. First, while existing event coreference resolvers have largely focused on exploiting the information extracted from the local contexts of the event mentions under consideration, we show how a state-of-the-art joint event coreference model can be improved using the *non-local* information provided by a supervised topic model and salient discourse entities. Second, the resulting model achieves the best results to date on the KBP 2017 English and Chinese event coreference datasets.

## 2 Definitions and Corpora

### 2.1 Definitions

We employ the following definitions in our discussion of trigger detection and event coreference:

- An **event trigger** is a string of text that most clearly expresses the occurrence of an event, usually a word or a multi-word phrase.
- An **event mention** is an explicit occurrence of an event consisting of a textual trigger, arguments or participants (if any), and the event type/subtype.
- An **event coreference chain** (a.k.a. an **event hopper**) is a group of event mentions that refer to the same real-world event. They must have the same event (sub)type.

To understand these definitions, consider the example in Table 1, which contains five event mentions from two documents. The first one consists of three event mentions of subtype `Personnel.Endposition`, among which  $ev1$  and  $ev2$ , which are triggered by “leave” and “departures” respectively, are coreferent since they describe the event that three journalists resign. The second one consists of two coreferent event mentions,  $ev4$  and  $ev5$ , both of which are triggered by “leave” and have subtype `Movement.Transport_Person`.

### 2.2 Corpora

We employ the English and Chinese corpora used in the TAC KBP 2017 Event Nugget Detection and Coreference task for evaluation, which are composed of two types of documents, newswire documents and discussion forum documents. There are no official training sets: the task organizers have simply made available a number of event coreference-annotated corpora for training. For English, we use LDC2015E29, E68, E73, E94, and LDC2016E64 for training. Together they contain 817 documents with 22894 event mentions distributed over 13146 coreference chains. For Chinese, we use LDC2015E78, E105, E112, and LDC2016E64 for training. Together they contain 548 documents with 7388 event mentions distributed over 5526 coreference chains.

The KBP 2017 English test set consists of 167 documents with 4375 event mentions distributed over 2963 coreference chains. The Chinese test set consists of 167 documents with 3884 event mentions distributed over 2558 coreference chains.

### 3 Model

Following Lu and Ng (2017a), we employ a structured conditional random field, which operates at the document level. Specifically, given a test document, we first extract from it all single- and multi-word nouns and verbs that have appeared at least once as a trigger in the training data. We treat each of these extracted nouns and verbs as a candidate event mention. The goal of the model is to make joint predictions for the candidate event mentions in a document. Three predictions will be made for each candidate event mention that correspond to the three tasks in the model: its trigger subtype, its induced topic, and its antecedent.

Given this formulation, we define three types of output variables. The first type consists of event subtype variables  $\mathbf{s} = (s_1, \dots, s_n)$ . Each  $s_i$  takes a value in the set of the 18 event subtypes defined in KBP 2017 or NONE, which indicates that the event mention is not a trigger. The second type consists of coreference variables  $\mathbf{c} = (c_1, \dots, c_n)$ , where  $c_i \in \{1, \dots, i - 1, \text{NEW}\}$ . In other words, the value of each  $c_i$  is the id of its antecedent, which can be one of the preceding event mentions, or NEW (if the mention underlying  $c_i$  starts a new cluster). The third type consists of topic variables  $\mathbf{t} = (t_1, \dots, t_n)$ . Each  $t_i$  takes a value in a 19-element set in which the topics have a one-to-one correspondence with the event subtype labels defined above. Despite this one-to-one mapping, these two types of labels should not be interpreted in the same manner. As we will see, a word’s induced topic label is influenced by our supervised topic model, whereas a word’s subtype is not.

Each candidate event mention is associated with one coreference variable, one event subtype variable, and one topic variable. Our model induces a probability distribution over these variables:

$$p(\mathbf{s}, \mathbf{c}, \mathbf{t} | x; \Theta) \propto \exp\left(\sum_i \theta_i f_i(\mathbf{s}, \mathbf{c}, \mathbf{t}, x)\right)$$

where  $\theta_i \in \Theta$  is the weight associated with feature function  $f_i$  and  $x$  is the input document.

#### 3.1 Independent Models

##### 3.1.1 Trigger Detection Model

Each instance for training the trigger detection model corresponds to a candidate trigger in the training set, which is created as follows. For each word  $w$  that appears as a true trigger at least once in the training data, we create a candidate trigger

from each occurrence of  $w$  in the training data. If a given occurrence of  $w$  is a true trigger in the associated document, the class label of the corresponding training instance is its subtype label. Otherwise, we label the instance as NONE.

Each candidate trigger  $m$  is represented using features generated from the following feature templates:  $m$ ’s word,  $m$ ’s lemma, word bigrams formed with a window size of three from  $m$ ; feature conjunctions created by pairing  $m$ ’s lemma with each of the following features: the head word of the entity syntactically closest to  $m$ , the head word of the entity textually closest to  $m$ , the entity type of the entity that is syntactically closest to  $m$ , and the entity type of the entity that is textually closest to  $m$ .<sup>1</sup> In addition, for event mentions with verb triggers, we use the head words and the entity types of their subjects and objects as features, where the subjects and objects are extracted from the dependency parses produced by Stanford CoreNLP (Manning et al., 2014). For event mentions with noun triggers, we create the same features except that we replace the subjects and verbs with heuristically extracted agents and patients.

##### 3.1.2 Topic Model

Our first extension to Lu and Ng’s (2017a) model seeks to improve trigger detection using topic information. We train a supervised topic model to infer the topic of each word in a test document, with the goal of understanding each candidate trigger using its *global* in addition to local context.

Like the trigger detection model, each training instance corresponds to a candidate trigger. The class label is the topic label of the candidate trigger. We have 19 topic labels in total: there is a one-to-one correspondence between the 18 subtype labels and 18 of the topic labels. The remaining topic label is OTHER, which is reserved for those words that do not belong to any of the 18 topics. Topic labels can be derived directly from subtype labels given the one-to-one correspondence between them. Each candidate trigger is represented using 19 features, which correspond to the 19 topic labels. The value of a feature, which is derived from the output of a LabeledLDA model (Ramage et al., 2009), encodes the probability that the candidate trigger belongs to the corresponding topic.

To train the LabeledLDA model, we first apply LabeledLDA using the Mallet toolkit (McCallum,

<sup>1</sup>We use an in-house CRF-based entity extraction model to jointly identify the entity mentions and their types.

2002) to the training documents, which learns a distribution over words for each topic,  $\beta$ . We represent each training document using the candidate triggers as well as the context words that are useful for distinguishing the topics.<sup>2</sup> To get the useful context words, we rank the words in the training documents by their weighted log-likelihood ratios:

$$P(w_i|m_j, v_k) \log \frac{P(w_i|m_j, v_k)}{P(w_i|m_j, \neg v_k)}$$

where  $w_i$ ,  $m_j$  and  $v_k$  denote the  $i$ th word in the vocabulary, the  $j$ th candidate trigger word and the  $k$ th subtype (including NONE), respectively. Intuitively, a word  $w_i$  will have a high rank with respect to a candidate trigger word  $m_j$  of subtype  $v_k$  if it appears frequently with  $m_j$  of subtype  $v_k$  and infrequently with  $m_j$  of other subtypes. We employ as the useful context words the top 125 words ranked by the weighted log likelihood ratio w.r.t. each pair of trigger and subtype. The label set of each training document is the set of subtypes collected from all the triggers in the document plus NONE.

After training, we apply the resulting LabeledLDA model to a test document, which is represented using the candidate triggers and the useful context words, as defined above. Specifically, given a test document, we (1) apply the model to infer the distribution of topics in the document, and then (2) compute the posterior distribution of topics given each candidate trigger in the document using Bayes rule as follows:

$$P(z|m) \propto P(m|z : \beta)P(z)$$

where  $P(z)$  is the distribution of topic  $z$  in the test document,  $P(m|z : \beta)$  is the topic-dependent distribution of candidate triggers  $m$  that is learned from the training documents, and  $P(z|m)$  is the posterior distribution of  $z$  given  $m$  in the test document. We use this posterior distribution to generate features for representing each instance for training/testing the topic model, as described above.

Note that while the label sets used by the trigger detector and the topic model are functionally equivalent, they are trained using different feature sets. The features used by the trigger detector encodes a candidate trigger’s local context, while the features used by the topic model encodes its global context (e.g., its relationship with other words).

---

<sup>2</sup>If a candidate trigger is a multi-word phrase, we treat it as a “word” by concatenating its constituent words using underscores (e.g., “step down” is represented as “step\\_down”).

### 3.1.3 Event Coreference Model

Our event coreference model is an adaptation of Durrett and Klein’s (2013) mention-ranking model, which was originally developed for entity coreference, to the task of event coreference. This model selects the most probable antecedent for a mention to be resolved from its set of candidate antecedents (or NEW if the mention is non-anaphoric).

We employ two types of feature templates to represent the candidate antecedents for the event mention to be resolved,  $m_j$ . The first type is composed of features that represent the NULL candidate antecedent.<sup>3</sup> These include:  $m_j$ ’s word,  $m_j$ ’s lemma, a conjoined feature created by pairing  $m_j$ ’s lemma with the number of sentences preceding  $m_j$ , and another conjoined feature created by pairing  $m_j$ ’s lemma with the number of mentions preceding  $m_j$  in the document. The second type is composed of features that represent a non-NUL candidate antecedent,  $m_i$ . These include  $m_i$ ’s word,  $m_i$ ’s lemma, whether  $m_i$  and  $m_j$  have the same lemma, and the following feature conjunctions: (1)  $m_i$ ’s word paired with  $m_j$ ’s word, (2)  $m_i$ ’s lemma paired with  $m_j$ ’s lemma, (3) the sentence distance between  $m_i$  and  $m_j$  paired with  $m_i$ ’s lemma and  $m_j$ ’s lemma, (4) the mention distance between  $m_i$  and  $m_j$  paired with  $m_i$ ’s lemma and  $m_j$ ’s lemma, (5) a quadruple consisting of  $m_i$  and  $m_j$ ’s subjects and their lemmas, and (6) a quadruple consisting of  $m_i$  and  $m_j$ ’s objects and their lemmas.

Our second extension to Lu and Ng’s (2017a) model involves leveraging *discourse* information to improve this event coreference model. Specifically, we introduce a *preprocessing* component for event coreference resolution where we *prune* the candidate antecedents of an event mention that are *unlikely* to be its correct antecedent based on *discourse context*. The idea is to (1) encode the discourse context of each event mention in a document using the *entities* that are *salient* at the point of the discourse in which the event mention appears, and by hypothesizing that two event mentions that appear in different discourse contexts are unlikely to be coreferent, we (2) prune any candidate antecedent of an event mention  $m$  whose discourse context is different from that of  $m$ , allowing the event coreference model to resolve an event mention to one of the candidate antecedents that survive this discourse-based filtering step. In essence, this

---

<sup>3</sup>Resolving a mention to the NULL antecedent is the same as having the mention starts a NEW cluster.

preprocessing step seeks to simplify the job of the event coreference model by reducing the number of candidate antecedents it has to consider for a given event mention.

Since we aim to encode the discourse context of each event mention using the entities that are salient at the point of the discourse in which the event mention appears, we need to compute the salience score of each entity  $E$  w.r.t. each event mention  $m$ . We employ the following formula, which was proposed by Chen and Ng (2015b):

$$\sum_{e \in E} g(e) \times \text{decay}(e)$$

In this formula,  $e$  is a mention of entity  $E$  that appears in either the same sentence as  $m$  or one of its preceding sentences.  $g(e)$  is a score that is computed based on the grammatical role of  $e$  in the sentence: 4 if  $e$  is a subject, 2 if it is an object, and 1 otherwise.  $\text{decay}(e)$  is a decay factor that is set to  $0.5^{\text{dis}}$ , where  $\text{dis}$  is the sentence distance between  $e$  and  $m$ . We compute discourse entities using Stanford CoreNLP’s neural entity coreference resolver and grammatical roles using CoreNLP’s syntactic dependency parser.

Next, we define the discourse context of an event mention  $m$  to be the list of entities whose salience score is at least 1 when computed w.r.t.  $m$ . As noted before, we aim to prune the unlikely candidate antecedents of an event mention  $m$ , namely those candidates whose discourse contexts are different from that of  $m$ . Rather than heuristically defining a function for computing the similarity between two different discourse contexts, we train a ranker that ranks the candidate antecedents of  $m$  based on two types of features derived from their discourse contexts:

**Salience score ratios (SSRs):** For each entity  $E$  that appears in the discourse contexts of both candidate antecedent  $c$  and  $m$ , we first compute  $E$ ’s SSR as the ratio of  $E$ ’s salience score computed w.r.t.  $m$  to  $E$ ’s salience score computed w.r.t.  $c$ . (If this ratio is less than 1, we take its reciprocal.) Then, for each  $(c, m)$  pair, we create five features that encode the number of entities whose SSR falls into each of these five intervals:  $[1, 1]$ ,  $(1, 2]$ ,  $(2, 3]$ ,  $(3, 4]$ ,  $(4, 5]$ , and  $[5, \inf]$ . Intuitively,  $c$ ’s and  $m$ ’s discourse contexts tend to be more similar if they have more entities in the lower buckets.

**Lexical features:** For each mention  $em_1$  of each entity in candidate antecedent  $c$ ’s discourse con-

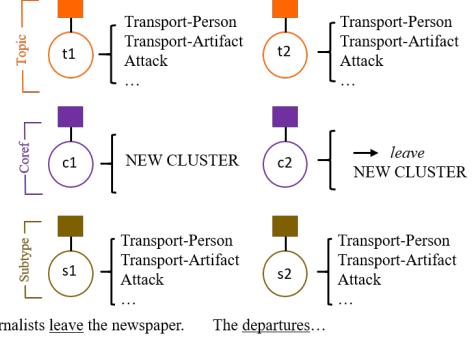


Figure 1: Unary factors for the three tasks, the variables they are connected to, and the possible values of the variables.

text and each mention  $em_2$  of each entity in  $m$ ’s discourse context, we create a lexical feature that pairs  $em_1$ ’s head with  $em_2$ ’s head.

To train this ranker, we employ the same log-linear model as the one used for the event coreference model, where the training objective is to maximize the likelihood of selecting the correct antecedent for each event mention.

After training, we apply this ranker to prune all but the top  $k$  candidate antecedents of each event mention in a test document. These  $k$  candidate antecedents, together with the NULL candidate antecedent, will be ranked by the event coreference model, and the highest-ranked candidate will be selected as the antecedent of the event mention under consideration.<sup>4</sup> We treat  $k$  as a hyperparameter and tune it on the development set.

It is worth noting that we prune the candidate antecedents of the event mentions not only in the test set but also in the training set. We produce the top  $k$  candidate antecedents of each event mention in the training set via five-fold cross-validation over the training documents.

Figure 1 illustrates the unary factors, which encode the features used in the three independent models. Specifically, the sentence fragment at the bottom of the figure contains two event mentions, one triggered by *leave* and the other by *departure*. Each of them is associated with three variables, one for each of the three models. Next to each variable is the set of possible values of that variable.

### 3.2 Joint Learning

To perform joint training over the three models described in the previous subsection, we need to

<sup>4</sup>The discourse preprocessing module does not handle NULL candidate antecedents, so they will always be available to the event coreference model.

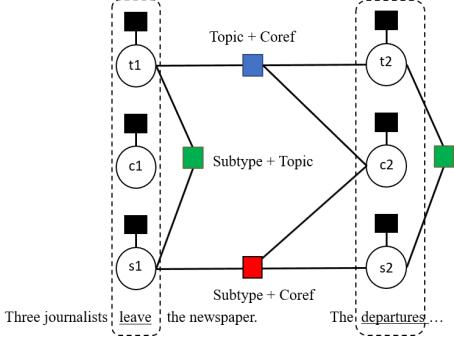


Figure 2: Binary and ternary factors.

define (1) features that capture the interaction between the two tasks, (2) the joint training scheme, and (3) the inference mechanism.

### 3.2.1 Cross-Task Interaction Features

Our cross-task interaction features, which capture the *pairwise* interaction between our tasks, are associated with ternary factors, as described below.

**Trigger detection and coreference.** We define our joint coreference and trigger detection factors such that the features defined on subtype variables  $s_i$  and  $s_j$  are fired only if current mention  $m_j$  is coreferent with preceding mention  $m_i$ . These features are: (1) the pair of  $m_i$  and  $m_j$ 's subtypes; (2) the pair of  $m_j$ 's subtype and  $m_i$ 's word; and (3) the pair of  $m_i$ 's subtype and  $m_j$ 's word.

**Trigger detection and topic modeling.** We fire features (encoded as binary factors) that conjoin each candidate event mention's event subtype, its topic and the lemma of its trigger.

**Topic modeling and coreference.** Our joint coreference and topic modeling factors and features are the same as those for trigger detection and coreference, except that event subtype labels are replaced with topic labels. In other words, the features are defined on the topic labels.

Figure 2 shows the cross-task interaction features. The green factor is binary, connecting a subtype variable and a topic variable. The red factor is ternary, connecting two subtype variables to a coreference variable. Finally, the blue factor is also ternary, connecting topic with coreference.

### 3.2.2 Training

The joint training scheme seeks to learn the model parameters  $\Theta$  from a set of  $d$  training documents, where document  $i$  contains content  $x_i$ , gold trigger annotations  $s_i^*$ , topic labels  $t_i^*$  inferred from the LabeledLDA model using Gibbs sampling, and

gold event coreference partition  $C_i^*$ , by maximizing the following conditional likelihood of the training data with  $L_1$  regularization:<sup>5</sup>

$$L(\Theta) = \sum_{i=1}^d \log \sum_{\mathbf{c}^* \in A(C_i^*)} p'(\mathbf{s}_i^*, \mathbf{t}_i^*, \mathbf{c}^* | x_i; \Theta) + \lambda \|\Theta\|_1$$

where  $p'(\mathbf{s}^*, \mathbf{t}^*, \mathbf{c}^* | x; \Theta)$  is  $p(\mathbf{s}^*, \mathbf{t}^*, \mathbf{c}^* | x; \Theta)$  augmented with task-specific loss functions. Specifically,

$$p'(\mathbf{s}^*, \mathbf{t}^*, \mathbf{c}^* | x; \Theta) \propto p(\mathbf{s}^*, \mathbf{t}^*, \mathbf{c}^* | x; \Theta) \exp[\alpha_s l_s(\mathbf{s}, \mathbf{s}^*) + \alpha_t l_t(\mathbf{t}, \mathbf{t}^*) + \alpha_c l_c(\mathbf{c}, C^*)]$$

where  $l_s$ ,  $l_t$  and  $l_c$  are task-specific loss functions<sup>6</sup>, and  $\alpha_s$ ,  $\alpha_t$  and  $\alpha_c$  are the associated weight parameters that specify the relative importance of the three tasks in the objective function.<sup>7</sup> We use AdaGrad (Duchi et al., 2011) to optimize our objective function with  $\lambda = 0.001$ .

### 3.2.3 Inference

Inference, which is performed during training and decoding, involves computing the marginals for a variable or a set of variables to which a factor connects. For efficiency, we perform approximate inference using belief propagation, running it until convergence. We use minimum Bayes risk decoding, where we compute the marginals for each variable in our model and independently return the most likely setting of each variable. Marginals typically converge in 3–5 iterations of belief propagation, so we use 5 iterations in our experiments.

## 4 Evaluation

### 4.1 Experimental Setup

We perform training and evaluation on the KBP 2017 English and Chinese corpora. For English,

<sup>5</sup>In the conditional log likelihood function,  $A(C_i^*)$  is the set of antecedent structures that are consistent with  $C_i^*$ . Since our model needs to be trained on antecedent vectors  $\mathbf{c}^*$  but the gold coreference annotation for each document  $i$  is provided in the form of a clustering  $C_i^*$ , we need to sum over all consistent antecedent structures.

<sup>6</sup>The loss function for event coreference, which is introduced by Durrett and Klein (2013) for entity coreference resolution, is a weighted sum of (1) the number of anaphoric mentions misclassified as non-anaphoric, (2) the number of non-anaphoric mentions misclassified as anaphoric, and (3) the number of incorrectly resolved mentions. The loss function for trigger detection is parameterized in a similar way, having three parameters associated with (1) the number of non-triggers misclassified as triggers, (2) the number of triggers misclassified as non-triggers, and (3) the number of triggers labeled with the wrong subtype. The loss function for topic detection is defined in a similar way as trigger detection.

<sup>7</sup>These weight parameters, as well as those that are used within the loss functions, are tuned on the development set using grid search.

		Event Coreference						Trigger Detection			
English		MUC	$B^3$	CEAF <sub>e</sub>	BLANC	AVG-F	$\Delta$	P	R	F	$\Delta$
1	Huang et al. (2019)	35.7	43.2	40.0	32.4	36.8		56.8	46.4	51.1	
2	Full	37.11	44.49	40.03	29.93	<b>37.89</b>		64.45	46.92	<b>54.30</b>	
3	– Topic	34.16	43.76	40.78	28.20	36.72	-1.17	64.39	46.67	54.11	-0.19
4	– Discourse	34.53	43.06	40.07	27.95	36.40	-1.49	62.15	47.49	53.84	-0.46
5	– Both	31.94	42.84	40.21	26.49	35.37	-2.52	63.57	45.87	53.29	-0.89
		Event Coreference						Trigger Detection			
Chinese		MUC	$B^3$	CEAF <sub>e</sub>	BLANC	AVG-F	$\Delta$	P	R	F	$\Delta$
6	Lu and Ng (2017b)	27.07	34.18	32.22	18.57	28.01		46.61	46.91	46.76	
7	Full	27.89	40.95	39.49	22.00	<b>32.58</b>		51.81	54.81	<b>53.27</b>	
8	– Topic	26.39	40.43	38.75	21.18	31.69	-0.89	51.81	53.28	52.53	-0.74
9	– Discourse	26.13	40.78	39.31	21.02	31.81	-0.77	51.65	54.65	53.11	-0.16
10	– Both	25.93	37.50	34.24	19.92	29.40	-3.18	56.78	44.63	49.98	-3.29

Table 2: Results of event coreference and trigger detection on the KBP 2017 English and Chinese test sets. Baseline results (rows 1 and 6) are copied verbatim from the original papers.

we train models on 646 of the training documents, tune parameters on 171 training documents, and report results on the official KBP 2017 English test set. For Chinese, we train models on 438 of the training documents, tune parameters on 110 training documents, and report results on the official KBP 2017 Chinese test set.

Results of event coreference and trigger detection are obtained using version 1.8 of the official scorer provided by the KBP 2017 organizers. To evaluate event coreference performance, the scorer employs four commonly-used scoring measures, namely MUC (Vilain et al., 1995),  $B^3$  (Bagga and Baldwin, 1998), CEAF<sub>e</sub> (Luo, 2005) and BLANC (Recasens and Hovy, 2011), as well as the unweighted average of their F-scores (AVG-F). The scorer reports event mention detection performance in terms of Precision (P), Recall (R) and F-score, considering a mention correctly detected if it has an exact match with a gold mention in terms of boundary and event subtype.

## 4.2 Results

Results on the English test set are shown in the top half of Table 2. Specifically, row 1 shows the results of Huang et al.’s (2019) resolver, which has produced best results to date on this test set. Row 2 shows the results of our full model, which substantially outperforms the baseline system (row 1), yielding an improvement of 1.09 points in AVG-F for event coreference and 3.2 points in F-score for trigger detection. Note that the improvement in the MUC and  $B^3$  F-scores is largely offset by the precipitation in the BLANC F-score.

Results on the Chinese test set are shown in the bottom half of Table 2. Specifically, row 6 shows the results of Lu and Ng’s (2017b) resolver, which is the top KBP 2017 system for Chinese and has

produced the best results to date on this test set. Our full model (row 7) outperforms this baseline by 4.57 points in AVG-F for event coreference and 6.51 points in F-score for trigger detection. Despite the large improvement in AVG-F, the MUC F-score only increases by 0.82 points. Since MUC F-scores are computed solely based on coreference links, these results suggest that the improvement in AVG-F can largely be attributed to successful identification singleton clusters rather than successful identification of coreference links.

## 4.3 Model Ablations

To evaluate the importance of each of the two extensions in the full model, we perform ablation experiments. Rows 3–5 and rows 8–10 in Table 2 show the English and Chinese results obtained using models that are retrained after one or both of the extensions are removed from the full model. The changes in AVG-F as a result of the ablations are shown in the  $\Delta$  columns for both tasks.

Similar conclusions can be drawn from the ablation results for both languages. First, ablating each of the two extensions causes a drop in performance for both event coreference and trigger detection. These results suggest that topic modeling and discourse pruning are both useful for the two tasks. Second, ablating both extensions causes a more abrupt drop in performance than ablating one of the extensions. This implies that each extension is providing useful information for each task that cannot be provided by the other extension. Third, when both extensions are ablated, the resulting models still outperform the baselines for both tasks. Nevertheless, we can see that for English, discourse pruning contributes more to the performance of our full model than topic modeling, whereas the reverse is true for Chinese.

		English	Chinese
		Training	Test
1	Number of candidate event mentions to be resolved	52370	9494
2	Number of candidate antecedents <b>before</b> pruning	371718	48750
3	Number of candidate antecedents <b>after</b> pruning	119416	20956
4	Number (%) of anaphoric event mentions	4362 (8.3%)	914 (9.6%)
5	Number (%) of anaphoric event mentions whose correct antecedent are among the candidates <b>before</b> pruning	4317 (99.0%)	803 (87.8%)
6	Number (%) of anaphoric event mentions whose correct antecedent are among the candidates <b>after</b> pruning	3171 (72.7%)	670 (73.3%)
		39758	9918
		124292	26406
		83378	20109
		1713 (4.3%)	821 (8.3%)
		1671 (97.6%)	585 (71.3%)
		1610 (94.0%)	565 (68.8%)

Table 3: Statistics on salience-based candidate pruning.

#### 4.4 Analysis of Salience-Based Pruning

To gain insights into the effectiveness of discourse modeling in terms of pruning candidate antecedents, Table 3 shows some statistics on the candidate antecedents before and after applying pruning. Concretely, row 1 shows the total number of event mentions to be resolved in the English and Chinese training and test sets. For English, as we can see in rows 2–3, only 32.1% and 43.0% of the candidate antecedents remain in the training and test sets respectively after pruning. This can be attributed to the fact that we aggressively prune the candidate antecedents by allowing  $k$  (the number of top candidate antecedents that can survive the pruning for each event mention) to be in the range of 1 to 5 during parameter tuning.<sup>8</sup> Row 4 shows that among all event mentions to be resolved, only 8.3% of them are anaphoric. Row 5 shows that before pruning, the correct antecedent of almost all of the anaphoric event mentions in the training set is among the set of candidate antecedents, whereas the corresponding number on the test set is only 87.8% due to the presence of unseen event mentions. Row 6 shows that 72.7% and 73.3% of the correct antecedents on the training set and the test set survive the pruning, respectively. Similar trends can be observed for the Chinese datasets. Overall, these statistics shed light on why discourse-based pruning is beneficial: the percentage of correct antecedents that survive the pruning is far greater than the percentage of candidate antecedents that are pruned.

#### 4.5 Discussion

One thing that the reader may not be able to appreciate just by looking at the performance numbers in Table 2 is that our two extensions are starting to attack some of the non-trivial aspects of event

<sup>8</sup>The best  $k$  according to the development set is 2 for English and 3 for Chinese.

coreference that involve semantics and discourse, as opposed to those previous approaches that focus on low-level issues (e.g., string matching). For this reason, we will take a look at some of the errors addressed by our extensions below.

Let us first consider the kind of errors topic modeling allows us to address. Consider the first two sentences in Table 4, both of which contain the trigger candidate “struck”. While “struck” triggers a “Conflict.Attack” event in the first sentence, neither of its occurrences in the second sentence corresponds to a true trigger (and therefore their subtypes should both be NONE). Without topic modeling, the model predicts all occurrences of “struck” in these sentences as belonging to Conflict.Attack (and hence misclassifies the subtypes of  $m_2$  and  $m_3$ ). The reasons are that (1) “struck” is most frequently associated with “Conflict.Attack” in the training data, and (2) since the two sentences have a similar syntactic structure and contain entities of the same type, the model fails to identify their differences. In contrast, with topic modeling, our model correctly predicts the topic of the document in which the second example appears as Contact.Meeting. Since the model manages to learn that the subtype of “struck” should be NONE when the topic is Contact.Meeting and that its subtype should be “Conflict.Attack” when the topic is “Conflict.Attack”, it correctly predicts  $m_2$  and  $m_3$  as having subtype NONE and, as a result, it also correctly determines that they are not coreferent. In other words, by using global information encoded by the topic model, our model can distinguish between words that have different meanings in different contexts.

Next, consider the last example in Table 4, which aims to give the reader an idea of the usefulness of discourse-based pruning. In this example,  $m_4$ ,  $m_5$ , and  $m_8$  refer to the event of the French soldier being stabbed and are coreferent, whereas  $m_6$  and  $m_7$

---

A barrage of US missile  $\{\text{struck}\}_{m_1}$  Pakistan’s North Waziristan tribal district on Tuesday, killing at least 15 militants.  
President Vladimir Putin sent his condolences to U.S. President Barack Obama on Tuesday over the deadly tornado that  $\{\text{struck}\}_{m_2}$  Oklahoma City. The tornado  $\{\text{struck}\}_{m_3}$  the southern suburbs of the Oklahoma state capital Monday afternoon, killing at least 51 people and injuring at least 140 others.

---

The French police said they were continuing to search for the man responsible for  $\{\text{stabbing}\}_{m_4}$  a uniformed soldier in the neck Saturday evening. The soldier was  $\{\text{stabbed}\}_{m_5}$  in the back of the neck with a box cutter or short knife as he patrolled with two colleagues through the transport station of La Défense, a business area in a suburb of Paris. The police suggested that the deed may have been inspired by the  $\{\text{attack}\}_{m_6}$  on a British soldier in a London street Wednesday. A spokesman for the police union UNSA, Christophe Crépin, said there were similarities with the London  $\{\text{attack}\}_{m_7}$ . The case of the  $\{\text{wounded}\}_{m_8}$  soldier, Pfc. Cédric Cordier, 23, is being handled by France’s anti-terrorism court, officials said Sunday.

---

Table 4: Examples illustrating the usefulness of topic modeling and salience-based pruning.

refer to the attack on the British solider and form another coreference cluster. Without discourse-based pruning, the model mistakenly links  $m_8$  with  $m_7$  because they both have subtype “Conflict.Attack”. In contrast, discourse-based pruning ranks  $m_4$  and  $m_5$  higher than  $m_6$  and  $m_7$  in  $m_8$ ’s list of candidate antecedents, the reason being that  $m_4$ ,  $m_5$ , and  $m_8$  share the same entity (realized as “a uniformed soldier”, “The soldier”, and “the wounded soldier”) in their contexts. Since the model retains only the top two candidate antecedents for English,  $m_6$  and  $m_7$  are being pruned, and the model successfully resolves  $m_8$  to  $m_5$ .

## 5 Related Work

**Using topics and salience.** For event coreference, the notion of “topics” has thus far been exploited only for *cross-document* event coreference, where documents are clustered by topics so that no cross-document coreference links can be established between documents in different clusters (Lee et al., 2012; Choubey and Huang, 2017). These resolvers, unlike ours, are pipelined systems, meaning that topic detection can influence event coreference resolution but not the other way round. As for discourse salience, we are not aware of any event coreference work that attempts to explicitly model it, although one can argue that existing systems may have implicitly encoded it in a shallow manner via exploiting features that encode the distance between two event mentions (Liu et al., 2014; Cybulska and Vossen, 2015).

**Computing argument compatibility.** In addition to discourse-based pruning, candidate antecedents can be pruned based on how compatible the arguments of the two event mentions are. To capture argument compatibility, argument features have been extensively exploited. Basic features such as the number of overlapping arguments and the number of unique arguments, and a binary feature encoding whether arguments are conflicting

have been proposed (Chen et al., 2009; Chen and Ji, 2009; Chen and Ng, 2016). More sophisticated features based on different kinds of similarity measures have also been considered, such as the surface similarity based on Dice coefficient and the WuPalmer WordNet similarity between argument heads (McConky et al., 2012; Cybulska and Vossen, 2013; Araki et al., 2014; Krause et al., 2016). These features are computed using either the outputs of event argument extractors and entity coreference resolvers (Ahn, 2006; Chen and Ng, 2014, 2015a; Lu and Ng, 2016) or the outputs of semantic parsers (Bejan and Harabagiu, 2014; Yang et al., 2015; Peng et al., 2016), and therefore suffer from error propagation (see Lu and Ng (2018)). Several previous works proposed joint models to address this problem (Lee et al., 2012; Lu et al., 2016), while others utilized iterative methods to propagate argument information (Liu et al., 2014; Choubey and Huang, 2017) in order to alleviate this issue. Nevertheless, argument extraction remains a very challenging task, especially when the arguments do not appear in the same sentence as the trigger. Our discourse-based pruning method can be thought of as a way of approximating argument compatibility without performing argument extraction.

## 6 Conclusion

We incorporated non-local information into a state-of-the-art joint model for event coreference resolution via topic modeling and discourse-based pruning. The resulting model not only significantly outperforms the independent models but also achieves the best results to date on the KBP 2017 English and Chinese event coreference corpora.

## Acknowledgments

We thank the three anonymous reviewers for their detailed and insightful comments on an earlier draft of the paper. This work was supported in part by NSF Grants IIS-1528037 and CCF-1848608.

## References

- David Ahn. 2006. [The stages of event extraction](#). In *Proceedings of the COLING/ACL Workshop on Annotating and Reasoning about Time and Events*, pages 1–8.
- Jun Araki, Zhengzhong Liu, Eduard Hovy, and Teruko Mitamura. 2014. [Detecting subevent structure for event coreference resolution](#). In *Proceedings of the 9th International Conference on Language Resources and Evaluation*, pages 4553–4558.
- Amit Bagga and Breck Baldwin. 1998. [Algorithms for scoring coreference chains](#). In *Proceedings of the LREC Workshop on Linguistic Coreference*, pages 563–566.
- Cosmin Adrian Bejan and Sanda Harabagiu. 2014. [Unsupervised event coreference resolution](#). *Computational Linguistics*, 40(2):311–347.
- Chen Chen and Vincent Ng. 2014. [SinoCoreferencer: An end-to-end Chinese event coreference resolver](#). In *Proceedings of the 9th International Conference on Language Resources and Evaluation*, pages 4532–4538.
- Chen Chen and Vincent Ng. 2015a. [Chinese event coreference resolution: An unsupervised probabilistic model rivaling supervised resolvers](#). In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1097–1107.
- Chen Chen and Vincent Ng. 2015b. [Chinese zero pronoun resolution: A joint unsupervised discourse-aware model rivaling state-of-the-art resolvers](#). In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 320–326.
- Chen Chen and Vincent Ng. 2016. [Joint inference over a lightly supervised information extraction pipeline: Towards event coreference resolution for resource-scarce languages](#). In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, pages 2913–2920.
- Zheng Chen and Heng Ji. 2009. [Graph-based event coreference resolution](#). In *Proceedings of the 2009 Workshop on Graph-based Methods for Natural Language Processing*, pages 54–57.
- Zheng Chen, Heng Ji, and Robert Haralick. 2009. [A pairwise event coreference model, feature impact and evaluation for event coreference resolution](#). In *Proceedings of the International Workshop on Events in Emerging Text Types*, pages 17–22.
- Prafulla Kumar Choubey and Ruihong Huang. 2017. [Event coreference resolution by iteratively unfolding inter-dependencies among events](#). In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2124–2133.
- Agata Cybulska and Piek Vossen. 2013. [Semantic relations between events and their time, locations and participants for event coreference resolution](#). In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages 156–163.
- Agata Cybulska and Piek Vossen. 2015. [Translating granularity of event slots into features for event coreference resolution](#). In *Proceedings of the 3rd Workshop on EVENTS*, pages 1–10.
- John Duchi, Elad Hazan, and Yoram Singer. 2011. [Adaptive subgradient methods for online learning and stochastic optimization](#). *Journal of Machine Learning Research*, 12:2121–2159.
- Greg Durrett and Dan Klein. 2013. [Easy victories and uphill battles in coreference resolution](#). In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1971–1982.
- Yin Jou Huang, Jing Lu, Sadao Kurohashi, and Vincent Ng. 2019. [Improving event coreference resolution by learning argument compatibility from unlabeled data](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 785–795.
- Sebastian Krause, Feiyu Xu, Hans Uszkoreit, and Dirk Weissenborn. 2016. [Event linking with sentential features from convolutional neural networks](#). In *Proceedings of the 20th Conference on Computational Natural Language Learning*, pages 239–249.
- Heeyoung Lee, Marta Recasens, Angel Chang, Mihai Surdeanu, and Dan Jurafsky. 2012. [Joint entity and event coreference resolution across documents](#). In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 489–500.
- Zhengzhong Liu, Jun Araki, Eduard Hovy, and Teruko Mitamura. 2014. [Supervised within-document event coreference using information propagation](#). In *Proceedings of the 9th International Conference on Language Resources and Evaluation*, pages 4539–4544.
- Jing Lu and Vincent Ng. 2016. [Event coreference resolution with multi-pass sieves](#). In *Proceedings of the 10th International Conference on Language Resources and Evaluation*.
- Jing Lu and Vincent Ng. 2017a. [Joint learning for event coreference resolution](#). In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 90–101.

- Jing Lu and Vincent Ng. 2017b. UTD’s event nugget detection and coreference system at KBP 2017. In *Proceedings of the 2017 Text Analysis Conference*.
- Jing Lu and Vincent Ng. 2018. Event coreference resolution: A survey of two decades of research. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 5479–5486.
- Jing Lu, Deepak Venugopal, Vibhav Gogate, and Vincent Ng. 2016. Joint inference for event coreference resolution. In *Proceedings of the 26th International Conference on Computational Linguistics*, pages 3264–3275.
- Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of the Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 25–32.
- Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60.
- Andrew Kachites McCallum. 2002. Mallet: A machine learning for language toolkit. <http://www.cs.umass.edu/~mccallum/mallet>.
- Katie McConky, Rakesh Nagi, Moises Sudit, and William Hughes. 2012. Improving event coreference by context extraction and dynamic feature weighting. In *Proceedings of the 2012 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support*, pages 38–43.
- Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. Joint event extraction via recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 300–309.
- Haoruo Peng, Yangqiu Song, and Dan Roth. 2016. Event detection and co-reference with minimal supervision. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 392–402.
- Daniel Ramage, David Hall, Ramesh Nallapati, and Christopher D. Manning. 2009. Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 248–256.
- Marta Recasens and Eduard Hovy. 2011. BLANC: Implementing the Rand Index for coreference evaluation. *Natural Language Engineering*, 17(4):485–510.
- Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Proceedings of the Sixth Message Understanding Conference*, pages 45–52.
- Bishan Yang, Claire Cardie, and Peter Frazier. 2015. A hierarchical distance-dependent Bayesian model for event coreference resolution. *Transactions of the Association for Computational Linguistics*, 3:517–528.

# Neural RST-based Evaluation of Discourse Coherence

Grigorii Guz<sup>\*1</sup>, Peyman Bateni<sup>\*1,2</sup>, Darius Muglich<sup>1</sup>, Giuseppe Carenini<sup>1</sup>

University of British Columbia<sup>1</sup>, Inverted AI<sup>2</sup>

{g.guz@cs, pbateni@cs, darius.muglich@alumni, carenini@cs}.ubc.ca

## Abstract

This paper evaluates the utility of Rhetorical Structure Theory (RST) trees and relations in discourse coherence evaluation. We show that incorporating silver-standard RST features can increase accuracy when classifying coherence. We demonstrate this through our tree-recursive neural model, namely RST-Recursive, which takes advantage of the text’s RST features produced by a state of the art RST parser. We evaluate our approach on the Grammarly Corpus for Discourse Coherence (GCDC) and show that when ensembled with the current state of the art, we can achieve the new state of the art accuracy on this benchmark. Furthermore, when deployed alone, RST-Recursive achieves competitive accuracy while having 62% fewer parameters.

## 1 Introduction

Discourse coherence has been the subject of much research in Computational Linguistics thanks to its widespread applications (Lai and Tetreault, 2018). Most current methods can be described as either stemming from explicit representations based on the Centering Theory (Grosz et al., 1994), or deep learning approaches that learn without the use of hand-crafted linguistic features.

Our work explores a third research avenue based on the Rhetorical Structure Theory (RST) (Mann and Thompson, 1988). We hypothesize that texts of low/high coherence tend to adhere to different discourse structures. Thus, we pose that using even silver-standard RST features should help in separating coherent texts from incoherent ones. This stems from the definition of the coherence itself - as the writer of a document needs to follow specific rules for building a clear narrative or argument structure in which the role of each constituent of the document should be appropriate with respect

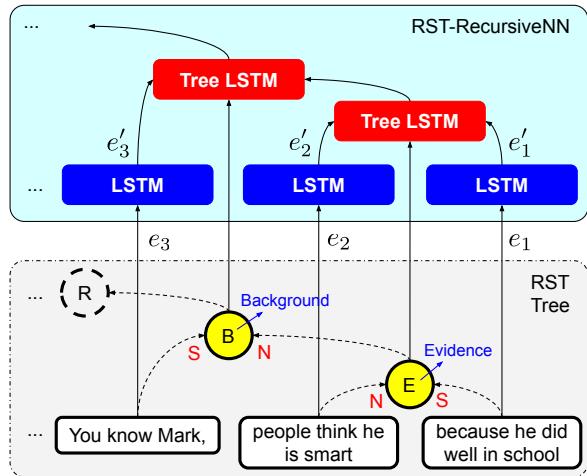


Figure 1: Overview of RST-Recursive; EDU embeddings are generated for the leaf nodes using the EDU network. Subsequently, the RST tree is recursively traversed bottom-up using the RST network.

to its local and global context, and even existing discourse parsers should be able to predict a plausible structure that is consistent across all coherent documents. However, if a parser has difficulty interpreting a given document, it will be more likely to produce unrealistic trees with improbable patterns of discourse relations between constituents. This idea was first explored by Feng et al. (2014), who followed an approach similar to Barzilay and Lapata (2008) by estimating entity transition likelihoods, but instead using discourse relations (predicted by a state of the art discourse parser (Feng and Hirst, 2014)) that entities participate in as opposed to their grammatical roles. Their method achieved significant improvements in performance even when using silver-standard discourse trees, showing potential in the use of parsed RST features for classifying textual coherence.

Our work, however, is the first to develop and test a neural approach to leveraging RST discourse representations in coherence evaluation. Furthermore, Feng et al. (2014) only tested their proposal on the

<sup>\*</sup>Authors contributed equally

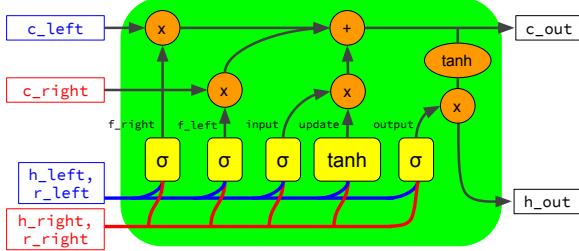


Figure 2: Recursive LSTM architecture used in RST-Recursive adapted from (Tai et al., 2015).

sentence permutation task, which involves ranking a sentence-permuted text against the original. As noted by Lai and Tetreault (2018), this is not an accurate proxy for realistic coherence evaluation. We evaluate our method on their more realistic Grammarly Corpus Of Discourse Coherence (GCDC), where the model needs to classify a naturally produced text into one of three levels of coherence. Our contributions involve: (1) RST-Recursive, an RST-based neural tree-recursive method for coherence evaluation that achieves 2% below the state of the art performance on the GCDC while having 62% fewer parameters. (2) When ensembled with the current state of the art, namely Parseq (Lai and Tetreault, 2018), we achieve a notable improvement over the plain ParSeq model. (3) We demonstrate the usefulness of silver-standard RST features in coherence classification, and establish our results as a lower-bound for performance improvements to be gained using RST features.

## 2 Related Work

### 2.1 Coherence Evaluation of Text

Centering Theory (Grosz et al., 1994) states that subsequent sentences in coherent texts are likely to continue to focus on the same entities (i.e., subjects, objects, etc.) as within the previous sentences. Building on top of this, Barzilay and Lapata (2008) were the first to propose the Entity-Grid model that constructs a two-dimensional array  $G_{n,m}$  for a text of  $n$  sentences and  $m$  entities, which are used to estimate transition probabilities for entity occurrence patterns. More recently, Elsner and Charniak (2011) extended Entity-Grid using entity-specific features, while Tien Nguyen and Joty (2017) used a Convolutional Neural Network (CNN) on top of Entity-Grid to learn more hierarchical patterns.

On the other hand, feature-free deep neural techniques have dominated recent research. Li and Jurafsky (2017) applied Recurrent Neural Networks (RNNs) to model the coherent generation of the

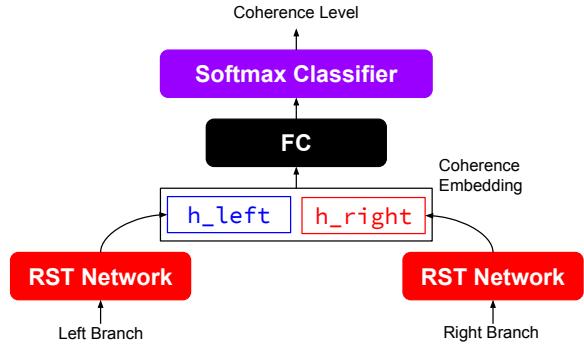


Figure 3: Overview of the classification layer in RST-Recursive; At the root of the RST tree, children’s hidden states are concatenated to form the document representation  $d = [h_l, h_r]$  which is then transformed into a 3-dimensional vector of Softmax probabilities.

next sentence given the current sentence and vice-versa. Mesgar and Strube (2018) constructed a local coherence model that encodes patterns of changes on how adjacent sentences within the text are semantically related. Recently, Moon et al. (2019) used a multi-component model to capture both local and global coherence perturbations. Lai and Tetreault (2018) developed a hierarchical neural architecture named ParSeq with three stacked LSTM Networks, designed to encode the coherence at sentence, paragraph and document levels.

### 2.2 Rhetorical Structure Theory (RST)

RST describes the structure of a text in the following way: first, the text is segmented into elementary discourse units (EDUs), which describe spans of text constituting clauses or clause-like units (Mann and Thompson, 1988). Second, the EDUs are recursively structured into a tree hierarchy where each node defines an RST relation between the constituting sub-trees. The sub-tree with the central purpose is called the *nucleus*, and the one bearing secondary intent is called the *satellite* while a connective discourse relation is assigned to both. An example of a “nucleus-satellite” relation pairing is presented in Figure 1 where a claim is followed by the evidence for the claim; RST posits an “Evidence” relation between these two spans with the left sub-tree being the “nucleus” and the right sub-tree as “satellite”.

## 3 Method

### 3.1 RST-Recursive

We parse silver-standard RST trees for documents using the CODRA (Joty et al., 2015) RST parser, which we then employ as input to our recursive neural model, RST-Recursive. The overall procedure

for RST-Recursive is shown in Figure 1. Given a document of  $n$  EDUs  $\mathcal{E}_{1:n}$  with each EDU  $\mathcal{E}_i$  represented as a list of GloVe embeddings (Pennington et al., 2014), we use an LSTM to process each  $\mathcal{E}_i$ , using the final hidden state as the EDU embedding  $\mathbf{e}_i = \text{LSTM}(\mathcal{E}_i)$  for each leaf  $i$  of the document’s RST tree. Afterwards, we apply a recursive LSTM architecture (Figure 2) that traverses the RST tree bottom-up. At each node  $s$ , we use the children’s sub-tree embeddings  $[\mathbf{h}_l, \mathbf{c}_l, \mathbf{r}_l]$  and  $[\mathbf{h}_r, \mathbf{c}_r, \mathbf{r}_r]$  to form the node’s sub-tree embedding:

$$[\mathbf{h}_s, \mathbf{c}_s] = \text{TreeLSTM}([\mathbf{h}_l, \mathbf{c}_l, \mathbf{r}_l], [\mathbf{h}_r, \mathbf{c}_r, \mathbf{r}_r]) \quad (1)$$

where  $\mathbf{h}_l/\mathbf{c}_l$  and  $\mathbf{h}_r/\mathbf{c}_r$  are the LSTM hidden and cell states from the left and right sub-trees respectively. The relation embeddings of the children sub-trees,  $\mathbf{r}_l$  and  $\mathbf{r}_r$ , are learned vector embeddings for each of the 31 pre-defined relation labels in the form of “[relation]\_[nucleus/satellite]” (e.g., “Evidence\_Satellite” for the last EDU in Figure 1). At the root of the tree, the output hidden states from both children are concatenated into a single document embedding  $\mathbf{d} = [\mathbf{h}_l, \mathbf{h}_r]$ . As shown in Figure 3, a fully connected layer is applied to this representation before using a Softmax function to obtain the coherence class probabilities.

### 3.2 Ensemble: ParSeq + RST-Recursive

To evaluate if the addition of silver-standard RST features to existing methods can improve coherence evaluation, we ensemble RST-Recursive with the current state of the art coherence classifier: ParSeq.

A deep learned non-linguistic classifier, ParSeq employs three layers of LSTMs that intend to capture coherence at different granularities. An overview of the ParSeq architecture is presented in Figure 4. First,  $\text{LSTM}_1$  (not shown) produces a single sentence embedding for each sentence in the text. Next,  $\text{LSTM}_2$  generates paragraph embeddings using the corresponding sentence embeddings from  $\text{LSTM}_1$ . Finally,  $\text{LSTM}_3$  reads the paragraph embeddings, generating the final document embedding, which is passed to a fully connected layer to produce Softmax label probabilities.

In this augmented variation of our model, we operate ParSeq on the document independently until a document level embedding  $\mathbf{d}_p$  is obtained at the highest-level LSTM. This document embedding is then concatenated to the RST-Recursive coherence embedding  $\mathbf{d} = [\mathbf{h}_l, \mathbf{h}_r, \mathbf{d}_{\text{parseq}}]$  in Figure

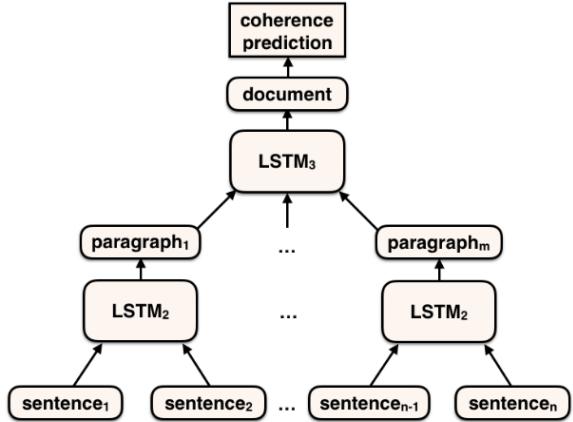


Figure 4: The architectural overview of ParSeq; an illustration of ParSeq’s structure, taken directly from the original paper (Lai and Tetreault, 2018).

3 to produce class probabilities. Note that in this ensemble variation, we initialize tree leaves  $\mathbf{e}_{1:n}$  with zero-vectors as opposed to EDU embeddings since ParSeq is sufficiently capable of capturing semantic information on its own, and early experiments using 5-fold cross-validation on the training set revealed model overfitting when training with EDU embeddings simultaneously.

## 4 Experiments

### 4.1 Dataset

We evaluate RST-Recursive and Ensemble on the GCDC dataset (Lai and Tetreault, 2018). This dataset consists of 4 separate sub-datasets: Clinton emails, Enron emails, Yahoo answers, and Yelp reviews, each containing 1000 documents for training and 200 documents for testing. Each document is assigned a discrete coherence label of incoherent (1), neutral (2), and coherent (3).

We parse RST trees for each example within the GCDC dataset using CODRA (Joty et al., 2015). Due to CODRA’s imperfect parsing of documents, RST trees could not be obtained for approximately 1.5%-2% of the documents, which were then excluded from the study. In addition, we re-evaluated ParSeq on only the RST-parsed portion of documents to assure consistent comparability of results. For more details, see Appendix A/B. Our code and dataset can be accessed below<sup>1</sup>, and the access to the original GCDC corpus can be obtained here<sup>2</sup>. We can share RST-parsings of GCDC examples with interested readers upon request once access to the GCDC dataset has also been obtained.

<sup>1</sup><https://github.com/grig-guz/coherence-rst>

<sup>2</sup><https://github.com/aylai/GCDC-corpus>

MODEL	T	NS	R	E	CLINTON	ENRON	YAHOO	YELP	AVERAGE
MAJORITY					55.33	44.39	38.02	54.82	48.14
RST-REC	✓				55.33±0.00	44.39±0.00	38.02±0.00	54.82±0.00	48.14±0.00
RST-REC	✓	✓			53.74±0.14	44.67±0.07	44.61±0.09	53.76±0.11	49.20±0.07
RST-REC	✓	✓	✓		54.07±0.10	43.99±0.07	49.39±0.10	54.39±0.12	50.46±0.05
RST-REC	✓	✓	✓	✓	55.70±0.08	53.86±0.11	50.92±0.13	51.70±0.16	53.04±0.09
PARSEQ					<b>61.05±0.13</b>	<b>54.23±0.10</b>	53.29±0.14	51.76±0.21	55.09±0.09
ENSEMBLE	✓			*	<b>61.12±0.13</b>	<b>54.20±0.12</b>	52.87±0.16	51.52±0.22	54.93±0.10
ENSEMBLE	✓	✓		*	60.82±0.13	54.01±0.10	52.92±0.15	51.63±0.24	54.85±0.10
ENSEMBLE	✓	✓	✓	*	<b>61.17±0.12</b>	53.99±0.10	<b>53.99±0.14</b>	<b>52.40±0.21</b>	<b>55.39±0.09</b>

Table 1: Overall and sub-dataset specific coherence classification accuracy on the GCDC dataset. Error boundaries describe 95% confidence intervals. Values in bold describe statistically significant state of the art performance. \* indicates availability of EDU-level semantic information through the ensembling with ParSeq.

MODEL	T	NS	R	E	CLINTON	ENRON	YAHOO	YELP	AVERAGE
MAJORITY					39.42	27.29	20.95	38.82	31.62
RST-REC	✓				39.42±0.00	27.29±0.00	20.95±0.00	38.82±0.00	31.62±0.00
RST-REC	✓	✓			39.20±0.03	30.81±0.16	35.67±0.18	39.93±0.08	36.40±0.09
RST-REC	✓	✓	✓		41.08±0.07	31.21±0.13	41.97±0.14	42.27±0.09	39.13±0.08
RST-REC	✓	✓	✓	✓	45.90±0.12	44.33±0.16	43.85±0.18	43.13±0.10	44.30±0.08
PARSEQ					<b>52.12±0.21</b>	<b>44.90±0.15</b>	46.22±0.18	43.36±0.09	46.65±0.10
ENSEMBLE	✓			*	<b>52.35±0.22</b>	<b>44.92±0.16</b>	45.48±0.22	43.70±0.11	46.61±0.11
ENSEMBLE	✓	✓		*	51.90±0.22	44.76±0.14	45.48±0.22	<b>43.83±0.13</b>	46.49±0.10
ENSEMBLE	✓	✓	✓	*	<b>52.42±0.19</b>	44.69±0.15	<b>46.88±0.17</b>	<b>43.94±0.09</b>	<b>46.98±0.09</b>

Table 2: Overall and sub-dataset specific coherence classification F1 scores on the GCDC dataset. Error boundaries describe 95% confidence intervals. Values in bold describe statistically significant state of the art performance. F1 scores are calculated by macro-averaging the corresponding class-wise F1 scores. \* indicates availability of EDU-level semantic information through the ensembling with ParSeq.

## 4.2 Training

We train all models with hyperparameter settings consistent with that of ParSeq reported by (Lai and Tetreault, 2018). Specifically, we use a learning rate of 0.0001, hidden size of 100, relation embedding size of 50, and 300-dimensional pre-trained GloVe embeddings (Pennington et al., 2014). We train with the Adam optimizer (Kingma and Ba, 2014) for 2 epochs. For every model/variation, the reported results represent the corresponding accuracies and F1 scores averaged over 1000 independent runs, each initialized with a different random seed.

## 4.3 RST-Recursive’s Performance

Our full model incorporates the RST Tree (T) structure, nucleus/satellite properties (nuclearity) of subtrees (NS), RST specific connective relations (R), and EDU embeddings at leaves of the RST tree (E),

as previously described in 3.1. Here, (T) defines the tree traversal operation and (NS) and (R) are learned vector embeddings for nuclearity and relations. We examine three ablations, each removing one of (NS), (R) and (E) from the model.

The results are provided in Tables 1 and 2. As shown, the complete model is able to achieve a competitive overall accuracy and F1 at 53.04% and 44.30% respectively, which is close to the state of the art. Although this lags behind ParSeq by a noticeable 2% margin, RST-Recursive is able to achieve this performance with 62% fewer parameters (1,230k vs. 3,241k), demonstrating the usefulness of linguistically-motivated features. Removing EDU embeddings reduces accuracy and F1 scores to 50.46% and 39.13%. This is still significantly better than the majority class baseline, signifying that even without any semantic infor-

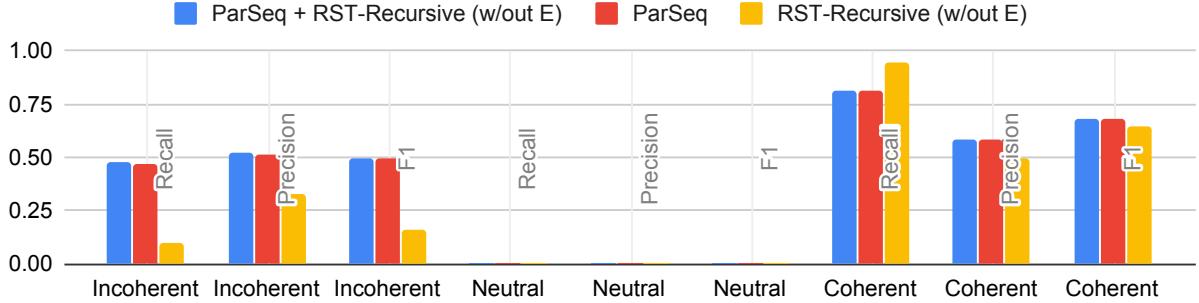


Figure 5: Comparison of Recall, Precision and F1 on overall classification of each coherence level.

mation about the text and its contents, it is still possible to evaluate coherence using just the silver-standard RST features of the text. Removing RST relations and nuclearity, however, decreases performance substantially, dropping to the majority class level. This indicates that an RST tree structure alone (of the quality delivered by silver-standard parsers) is not sufficient to classify coherence. It must also be noted that since we employ silver-standard RST parsing as performed by CODRA (Joty et al., 2015), the reported results act as a lower bound which we would expect to improve as parsing quality increases.

#### 4.4 Ensemble’s Performance

We examine three variations of the Ensemble. The full model augments ParSeq with the text’s RST tree, relations and nuclearity. This model is able to achieve the new state of the art performance, at 55.39% accuracy and 46.98% F1. Using final layer concatenation for ensembling is widely applicable to many other neural methods, and serves as a lower bound for the accuracy/F1 boost to be appreciated by incorporating RST features into the model. Removing the RST relations and/or nuclearity information completely eliminates the performance gain, which shows that the RST tree on its own is not sufficient as an RST source of information for distinguishing coherence, even when ensembled with ParSeq.

#### 4.5 Classification Trends

As demonstrated in Figure 5, coherence classifiers have difficulty predicting the neutral class (2), experiencing modal collapse towards the extreme ends in the best performing models. Early experiments using alternative objective functions such as the Ordinal Loss or Mean Squared Error resulted in a similar modal collapse or poor overall performance. We leave further exploration of this problem to future research. Furthermore, RST-

Recursive shows a notably stronger recall on the coherent class (3) as compared to ParSeq. On the other hand, ParSeq has a higher recall/precision on class (1) and slightly higher precision on class (3). The Ensemble method, however, is able to take the best of both, achieving better recall, precision and F1 on both the incoherent and coherent classes as compared to ParSeq.

## 5 Conclusions and Future Work

In this paper, we explore the usefulness of silver-standard parsed RST features in neural coherence classification. We propose two new methods, RST-Recursive and Ensemble. The former achieves reasonably good performance, only 2% short of state of the art, while more robust with 62% fewer parameters. The latter demonstrates the added advantage of RST features in improving classification accuracy of the existing state of the art methods by setting new state of the art performance with a modest but promising margin. This signifies that the document’s rhetorical structure is an important aspect of its perceived clarity. Naturally, this improvement in performance is bounded by the quality of parsed RST features and could increase as better discourse parsers are developed.

In the future, exploring other RST-based architectures for coherence classification, as well as better RST ensemble schemes and improving RST parsing can be avenues of potentially fruitful research. Additional research on multipronged approaches that draw from Centering Theory, RST and deep learning all together can also be of value.

## References

- Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34.
- Lynn Carlson, Mary Ellen Okurowski, and Daniel

- Marcy. 2002. Rst discourse treebank. *Linguistic Data Consortium, University of Pennsylvania*.
- Micha Elsner and Eugene Charniak. 2011. **Extending the entity grid with entity-specific features**. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 125–129, Portland, Oregon, USA. Association for Computational Linguistics.
- Vanessa Wei Feng and Graeme Hirst. 2014. **A linear-time bottom-up discourse parser with constraints and post-editing**. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 511–521, Baltimore, Maryland. Association for Computational Linguistics.
- Vanessa Wei Feng, Ziheng Lin, and Graeme Hirst. 2014. **The impact of deep hierarchical discourse structures in the evaluation of text coherence**. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 940–949, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.
- Barbara Grosz, Aravind Joshi, and Scott Weinstein. 1994. Centering: A framework for modelling the coherence of discourse. *Technical Reports (CIS)*.
- Shafiq Joty, Giuseppe Carenini, and Raymond Ng. 2015. **Codra: A novel discriminative framework for rhetorical analysis**. *Computational Linguistics*, 41:1–51.
- Diederik P. Kingma and Jimmy Ba. 2014. **Adam: A method for stochastic optimization**. Cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.
- Alice Lai and Joel R. Tetreault. 2018. **Discourse coherence in the wild: A dataset, evaluation and methods**. *CoRR*, abs/1805.04993.
- Jiwei Li and Dan Jurafsky. 2017. **Neural net models of open-domain discourse coherence**. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 198–209, Copenhagen, Denmark. Association for Computational Linguistics.
- William Mann and Sandra Thompson. 1988. **Rethorical structure theory: Toward a functional theory of text organization**. *Text*, 8:243–281.
- Mohsen Mesgar and Michael Strube. 2018. **A neural local coherence model for text quality assessment**. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4328–4339, Brussels, Belgium. Association for Computational Linguistics.
- Han Cheol Moon, Tasnim Mohiuddin, Shafiq Joty, and Chi Xu. 2019. **A unified neural coherence model**. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2262–2272, Hong Kong, China. Association for Computational Linguistics.
- Mathieu Morey, Philippe Muller, and Nicholas Asher. 2017. How much progress have we made on rst discourse parsing? a replication study of recent results on the rst-dt. In *EMNLP*.
- Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. **Glove: Global vectors for word representation**. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.
- Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Milt-sakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. 2008. The penn discourse treebank 2.0.
- Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. **Improved semantic representations from tree-structured long short-term memory networks**. *CoRR*, abs/1503.00075.
- Dat Tien Nguyen and Shafiq Joty. 2017. **A neural local coherence model**. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1320–1330, Vancouver, Canada. Association for Computational Linguistics.

# Appendices

## A Dataset Description

For model evaluation, we use the recently released Grammarly Corpus for Discourse Coherence (Lai and Tetreault, 2018). GCDC consists of 4 sections - Clinton and Enron emails, as well as Yelp review and Yahoo answers, with 1000 training and 200 testing examples in each section. Each text is given a score from 1 (least coherent) to 3 (most coherent) by expert raters. GCDC’s key advantage, compared to the ranking corpora used in the past (Prasad et al., 2008), is that all the datapoints are human-labelled and not artificially permuted. Examples from the

---

### Coherence / Example

---

#### Incoherent (1)

For good Froyo, you just got to love some MoJo, yea baby yea! Creamy goodness with half the guilt of ice cream, a spread of tasty toppings, this in the TMP in definitely the place to be! They have little cups for sampling to find your favorite flavor. Great prices and with a yelping good 25% off discount just for "checking in" and half off Tuesdays with the FB word of the day, you just can't beat it! Perfect summer treat located in front of the TMP splash pad, you can soak up some sun and enjoy some fromazing yogurt in their outdoor sitting area! Go get you some Mojo froyo!

---

#### Neutral (2)

So Spintastic gets 5 stars because it's about as good as it gets for a laundromat, me thinks.  
Came here bc the dryer at my place was busted and waiting on the repairman. I found the people working the place extremely helpful. It was my first time there and she walked me through the steps of how to get a card, which machines to use, where I could buy the soap... only thing she didn't do was fold my dried laundry! Heh. Will remember this place for the future in the event that I need to get my clothes washed and ready. Free wi-fi and a soda machine is convenient. Oh and if you have a balance left on your card, you can redeem the card and any remaining balance if you like.  
dmo out

---

#### Coherent (3)

vet for almost 6 years. He is kind, compassionate and very loving and gentle with my dogs. All my dogs are shelter dogs and I am very picky about who cares for my animals.  
I walked in once with a dog I found running around the neighborhood and the staff could not find a chip so Dr. Besemer came out to help. He was busy but made time for me. He looked over the dog and could not find a chip, he also did a quick check on the dog and said that he appeared healthy. He didn't charge me for his time. This dog became my third adopted dog. Dr. Besemer is the best and I highly recommend him if you are looking for a vet. His staff is kind and compassionate.

---

Table 3: Text examples of incoherent (class 1), neutral (class 2), and coherent (class 3) snippets from the Yelp subset of the GCDC dataset (Lai and Tetreault, 2018).

Parser	Structure	Nuclearity	Relation	Full
CODRA	82.6	68.3	55.8	55.4
Human	88.3	77.3	65.4	64.7

Table 4: Micro-averaged F1 scores on the RST parsing of text by CODRA vs. Human Standard (Morey et al., 2017).

dataset are provided in Table 3. When assigning the ranking to each text, the experts received the following instructions (Lai and Tetreault, 2018):

*A text that is highly coherent (score 3) is easy to understand and easy to read. This usually means the text is well-organized, logically structured, and presents only information that supports the main idea. On the other hand, a text with low coherence (score 1) is difficult to understand. This may be because the text is not well organized, contains unrelated information that distracts from the main idea, or lacks transitions to connect the ideas in the text. Try to ignore the effects of grammar or spelling errors when assigning a coherence rating.*

We generated a discourse tree for each text in the GCDC dataset, utilizing the available CODRA discourse parser (Joty et al., 2015). Early iterations resulted in up to 30% unsuccessful parsing rate on some sub-datasets. As a result, a punctuation fixing script was developed to fix minor punctuation problems without changing the text’s structure or coherence. Post-fixing results lowered this RST parsing failure rate to reasonable margins in the 1% to 3% region (see Table 5). Note that all examples for which RST parsing was not successfully performed were excluded in our experiments. All baselines were re-evaluated using the RST-parsed set of examples.

## B CODRA Quality

While partial parsing of the dataset (see Appendix A) allows us to evaluate the accuracy of our models, it must be emphasized that as with the goal of this paper, we’ve used silver-standard RST parsing which lags well behind the human gold-standard. As shown in Table 4, CODRA is far from reaching human-level accuracy in RST parsing. Additionally, since it was trained on RST-DT (Carlson et al., 2002), it lacks out-of-domain adaptability, which becomes a bottle-neck in achieving substantial performance boost on badly structured domains of text such Yelp review. We again re-iterate the importance of RST parsing for RST-based coherence evaluation, and motivate future work in this area.

	TRAIN				TEST			
	CLINTON	ENRON	YAHOO	YELP	CLINTON	ENRON	YAHOO	YELP
EXAMPLES	1000	1000	1000	1000	200	200	200	200
PRE-FIX RST-TREES	667	710	940	950	136	142	188	190
POST-FIX RST-TREES	985	976	986	999	199	195	192	197
POST-FIX VERY COHERENT	503	499	368	511	109	87	73	109
POST-FIX MEDIUM COHERENT	204	192	170	218	38	50	41	42
POST-FIX INCOHERENT	277	289	442	270	50	59	78	47

Table 5: Number of examples for which RST trees were successfully produced in each GCDC sub-dataset.

We believe that improvements in RST parsing will result in better accuracy for both future and existing RST-based coherence evaluation methods.

# Asking Crowdworkers to Write Entailment Examples: The Best of Bad Options

Clara Vania Ruijie Chen Samuel R. Bowman

New York University

{c.vania, rc3959, bowman}@nyu.edu

## Abstract

Large-scale natural language inference (NLI) datasets such as SNLI or MNLI have been created by asking crowdworkers to read a *premise* and write three new *hypotheses*, one for each possible semantic relationships (*entailment*, *contradiction*, and *neutral*). While this protocol has been used to create useful benchmark data, it remains unclear whether the writing-based annotation protocol is optimal for any purpose, since it has not been evaluated directly. Furthermore, there is ample evidence that crowdworker writing can introduce artifacts in the data. We investigate two alternative protocols which automatically create candidate (*premise*, *hypothesis*) pairs for annotators to label. Using these protocols and a writing-based baseline, we collect several new English NLI datasets of over 3k examples each, each using a fixed amount of annotator time, but a varying number of examples to fit that time budget. Our experiments on NLI and transfer learning show negative results: None of the alternative protocols outperforms the baseline in evaluations of generalization within NLI or on transfer to outside target tasks. We conclude that crowdworker writing still the best known option for entailment data, highlighting the need for further data collection work to focus on improving *writing-based* annotation processes.

## 1 Introduction

Research on natural language understanding has benefited greatly from the availability of large-scale, annotated data, especially for tasks like reading comprehension and natural language inference, which lend themselves to non-expert crowdsourcing. These datasets are useful in three settings: evaluation (Williams et al., 2018; Rajpurkar et al., 2018; Zellers et al., 2019); pretraining (Phang et al., 2018; Conneau et al., 2018; Pruksachatkun et al.,

2020); and as training data for downstream tasks (Trivedi et al., 2019; Portelli et al., 2020).

Natural language inference (NLI), also known as *recognizing textual entailment* (RTE; Dagan et al., 2005) is the problem of determining whether or not a hypothesis semantically entails a premise. The two largest NLI corpora, SNLI (Bowman et al., 2015) and MNLI (Williams et al., 2018) are created by asking crowdworkers to write three labeled *hypothesis* sentences given a *premise* sentence taken from a preexisting text corpus. While these datasets have been widely used as benchmarks for NLU, there have been no studies evaluating writing-based annotation for collecting NLI data. Moreover, there is growing evidence that human writing can introduce *annotation artifacts*, which enable models to perform moderately well just by learning spurious statistical patterns in the data (Gururangan et al., 2018; Tsuchiya, 2018; Poliak et al., 2018a).

This paper explores the possibility of collecting high-quality NLI data without asking crowdworkers to write hypotheses. We introduce two alternative protocols (Figure 1) which substitute crowdworker writing with fully-automated pipelines to generate premise-hypothesis sentence pairs, which annotators then simply label. The first protocol uses a sentence-similarity-based method to pair similar sentences from large unannotated corpora. The second protocol uses parallel sentences and uses machine translation systems to generate sentence pairs. Using the MNLI protocol as our baseline, we collect five datasets using premises taken from Gigaword news text (Parker et al., 2011) and Wikipedia. We then compare models trained using these datasets for their generalization performance within NLI and for transfer learning to other tasks.

We start from the assumption that writing a new hypothesis takes more time and effort than simply labeling a presented hypothesis. As a result, it is plausible that our protocols could offer some value

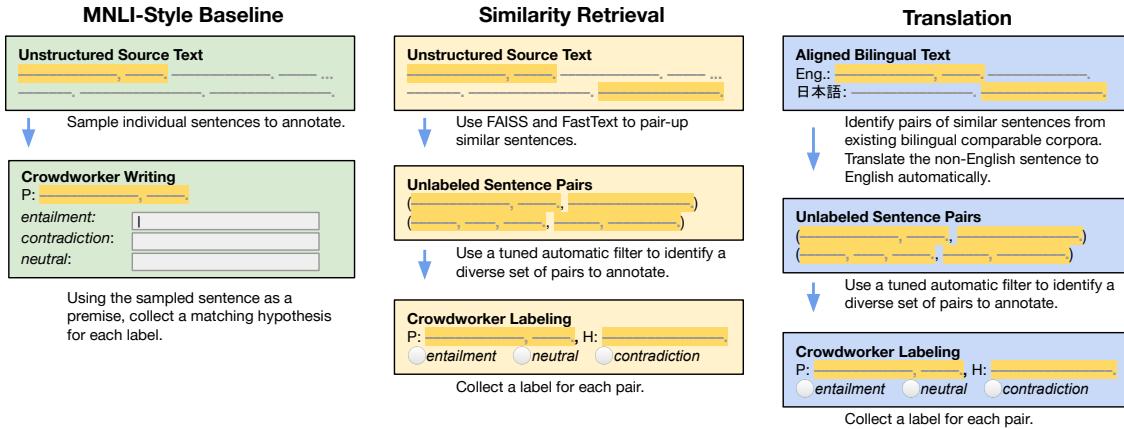


Figure 1: We introduce two new protocols for natural language inference data collection. Both use fully-automated pipelines to generate pairs of semantically-related sentences, which crowdworker annotators then label.

even if the quality of the data they produce is no better than a writing-based baseline. To study the cost trade-off, we collect each dataset under the same fixed annotation budget with a fixed ( $\sim$  US \$15) hourly wage. Using this constraint, we collect approximately twice as many examples from our new protocols.

Our main results on natural language inference and transfer learning are clearly negative. Human-constructed examples appear to be far superior to automatically-constructed examples in both settings. While crowdworker writing in data collection has known issues, it produces better training data than our automatic methods, or any known comparable methods which intervene the writing-based protocol to help crowdworkers with the writing process (Bowman et al., 2020). This strongly suggests that future work on data quality should focus on improving human-based generation processes.

## 2 Collecting NLI Data

We compare three protocols for collecting NLI data: (1) a baseline MNLI-style protocol (BASE), (2) a sentence-similarity-based protocol (SIM), and (3) a translation-based protocol (TRANSLATE). To test generalization performance across domains, we collect two datasets for BASE and SIM, using text from Gigaword (news) and Wikipedia (wiki) domains.<sup>1</sup> For TRANSLATE, we collect a dataset from WikiMatrix (Schwenk et al., 2019), a collection of Wikipedia parallel sentences. Table 1 shows examples of sentence pairs collected using

<sup>1</sup>The premise sentences for each protocol can be different although they come from the same source.

each protocol.

Our new protocols (Figure 1) share a similar automated pipeline. Given an unstructured text, we automatically collect similar sentence pairs which annotators then label. There are two key differences between our new protocols and BASE. First, our automatically paired sentences are *unlabeled*, and thus require a further data labeling process (Section 2.4). Second, our protocols might produce datasets with imbalanced label distributions. This is in contrast to BASE, which ensures each premise will have one hypothesis for each label in the annotation. The following subsections describe each protocol in more detail.

### 2.1 Baseline (BASE)

Our BASE protocol closely follows that used for MNLI. We randomly sample premise sentences from Gigaword and Wikipedia and ask crowdworkers to write three new hypotheses, one for each relation type.<sup>2</sup>

### 2.2 Sentence Similarity (SIM)

Our SIM protocol exploits the fact that, in large corpora, it should be easy to find pairs of sentences that describe similar events or situations. For example, in Gigaword, one event might be written differently by different news sources in ways that yield any of our three relationships. We collect similar sentences and automatically match them to form sentence pairs which annotators then label. The whole pipeline consists of three steps:

<sup>2</sup>Our instructions can be found in the Appendix A, and our FAQs are available at <https://sites.google.com/nyu.edu/nlu-mturk-faq/writing-sentences>.

Dataset	Label	Premise	Hypothesis
Base-News	E	The city reconsidered that position on Wednesday, saying it was seeking to raise an additional \$1.5 million to extend Mardi Gras over two weekends and to pay for overtime on several days.	The city is looking to get more money for Mardi Gras.
Base-Wiki	C	Service books were not included and a note at the end mentions many other books in French, English and Latin which were then considered worthless.	Service books were included.
Sim-News	N	All of them run out like college football players before a big bowl game.	Pray before a college football game.
Sim-Wiki	C	His work was heavily criticised as unscientific by his contemporaries.	His work was recognized and admired by his contemporaries.
Translate-Wiki	E	This was used to indicate a positive response, or truth, or approval of the item in front of it.	This was used to indicate yes, true, or confirmed on items in a list.

Table 1: Examples of sentence pairs chosen randomly from each test set, along with their assigned labels. E: *entailment*, C: *contradiction*, N: *neutral*.

indexing and retrieval, reranking, and crowdworker labeling.

**Indexing and Retrieval** Given a raw text, we first split it into sentences.<sup>3</sup> We encode each sentence as a 300-dimensional vector using fastText (Bojanowski et al., 2017) and index them using FAISS (Johnson et al., 2019), an open-source library for large-scale similarity search on vectors.<sup>4</sup> Since Gigaword and Wikipedia consist of billions of sentences, we perform dimensionality reduction using PCA and cluster the search space to allow efficient index and retrieval. We randomly sample query sentences from the text corpus and retrieve the top 1k most similar sentences for each query. This is done by building an index with type "PCAR64, IVFx, Flat" in FAISS terms, where x varies depending on the corpus size. Details of our indexing and retrieval procedures can be found in Appendix A.1.

**Reranking** FastText uses a Continuous Bag-of-Words (CBOW) model to learn word representations. This means given a query, we will sometimes have top matches which are syntactically similar but describe different events or situations. While unrelated sentences can be contradictory or neutral, directly using the top- $n$  sentences from FAISS will give us too few entailment pairs. Furthermore, because we use randomly sampled sentences as queries, there could be no good match at all for a given query.

<sup>3</sup>We use Spacy's "en\_core\_web\_lg" model to segment sentences and extract noun phrase and entities for later use in reranking.

<sup>4</sup><https://github.com/facebookresearch/faiss>

To collect a set of sentence pairs with a reasonable label distribution, for each query, we retrieve top- $K$  matches and rerank the (query, retrieved sentence) pairs using the following features:

- *FAISS similarity score*: The raw similarity score from FAISS.
- *Word types*: The proportion of word types in the query sentence seen in the retrieved sentence.
- *Noun phrase*: The proportion of noun phrases in the query sentence seen in the retrieved sentence.
- *Subjects*: The proportion of complete subject spans (some sentences with embedded clauses can have more than one subject) in the query sentence seen in the retrieved sentence.
- *Named entity*: The proportion of named entities in the query sentence seen in the retrieved sentence.
- *Time*: A boolean feature which denotes whether two sentences are written in the same month and year (only for Gigaword)
- *Wiki article*: A boolean feature which denotes whether the pairs come from the same article. (only for Wikipedia)
- *Wiki link*: The proportion of hyperlink tokens in the query sentence seen in the retrieved sentence (only for Wikipedia)

The choice of these hand-crafted features will likely impact the distribution of our final dataset, but we

don’t expect these choices to inject significant label-association artifacts, since our methods play no role in setting labels. We calculate the score for each sentence pair using a weighted sum of these features. We populate pairs from all queries and sort them based on their feature scores. We then select the top  $N\%$  pairs as our final pairs.

We use a Bayesian hyperparameter optimization to tune the feature weights,  $K$ , and  $N$ . In an ideal case, we want our dataset to have a balanced distribution so that all classes will be represented equally. To push for this, we tune these parameters to minimize the Kullback–Leibler (KL) divergence between a uniform distribution across three entailment classes,  $P(x)$ , and an empirical distribution,  $Q(x)$ , computed based on the predictions of an NLI model. We run Bayesian optimization for 100 iterations using Optuna (Akiba et al., 2019). For the NLI model, we use a RoBERTa<sub>Large</sub> model fine-tuned on a combination of SNLI, MNLI, and ANLI.

### 2.3 Translation (TRANSLATE)

Multilingual comparable corpora contain *similar* texts in at least two different languages. If they are sentence-aligned, we can automatically translate text from one language to one of the others to yield candidate sentence pairs. Since the alignment behind the corpus can be noisy, the resulting sentence pairs range almost continuously from being parallel to being semantically unrelated, potentially fitting any of the three entailment relationships. In the TRANSLATE protocol, we investigate whether we can use such sentence pairs as entailment data.

We use WikiMatrix (Schwenk et al., 2019), a collection of 135 million Wikipedia parallel sentences, which was constructed by aligning similar sentences in different languages in a joint sentence embedding space (Schwenk, 2018; Artetxe and Schwenk, 2019). It is a mix of translated sentence pairs and comparable sentences written independently about the same information. We collect parallel sentences where one of the sentences is in English,  $s^E$ . For the paired non-English languages, we pick 5 languages: German, French, Indonesian, Japanese, and Czech. We then translate the aligned non-English sentence into an English sentence,  $\hat{s}^E$  using the OPUS-MT (Tiedemann and Thottingal, 2020) machine translation systems, and treat  $(s^E, \hat{s}^E)$  as a sentence pair. The diverse set of languages allows us to collect a more diverse set

	<b>Individual == Gold</b>	<b>No Gold Label</b>
MNLI (Full)	88.7%	1.8%
Base-News	78.7%	13.1%
Base-Wiki	76.4%	10.0%
Sim-News	72.9%	15.8%
Sim-Wiki	74.1%	11.9%
Translate-Wiki	72.8%	14.6%

Table 2: Validation statistics for each protocol, compared to MNLI Full.

of sentence pairs coming from the structural differences across languages. We do not perform any reranking as our predictions using an NLI model on the initially retrieved data (the same one that we used in §2.2) shows a near-balanced distribution.

### 2.4 Data Labeling

We use Amazon Mechanical Turk to label the automatically-collected sentence pairs (SIM and TRANSLATE). We hire crowdworkers which have completed at least 5000 HITs with at least a 99% acceptance rate. In each task, we present crowdworkers with a sentence pair and ask them to provide a single label (*entailment*, *contradiction*, *neutral* or “*I don’t understand*”) for the pair. The latter is used if there are problems with either sentence, e.g., because of errors during preprocessing. We collect one label per sentence pair. We use the same HIT setup for validating our test sets (Section 3).

## 3 The Resulting Datasets

Using BASE, we collect 3k examples for Base-News and Base-Wiki.<sup>5</sup> For SIM and TRANSLATE, we increase the number of pairs to exhaust the same budget that was used for the corresponding baseline dataset (\$1,791 for Base-News and \$1,445 for Base-Wiki), allowing us to collect around twice as many examples for each protocol.<sup>6</sup>

For each dataset, we randomly select 250 sentence pairs as the test set and use the rest as the training set. To ensure accurate labeling, we perform an additional round of annotation on the test sets. We ask four crowdworkers to label each pair using the same instructions that we use for data labeling, giving us a total of 5 annotations per example. We assign the majority vote as the gold

<sup>5</sup>Our preliminary experiments on subsets of MNLI show that RoBERTa performance starts to stabilize once we use at least 3k training examples.

<sup>6</sup>The resulting datasets are available at <https://github.com/nyu-mll/semi-automatic-nli>. We provide anonymized worker-ids.

	#Pairs	Label Distribution			$\text{HL}_E$		$\text{HL}_C$		$\text{HL}_N$		Word Type Overlap			
		E	C	N	$\mu$	( $\sigma$ )	$\mu$	( $\sigma$ )	$\mu$	( $\sigma$ )	E	C	N	
Training	MNLI-3k	2750	33.4	33.9	32.7	9.7	4.4	9.4	4.0	11.0	4.4	25.2	17.3	15.4
	Base-News	2734	33.5	33.4	33.2	12.1	6.0	11.8	5.8	12.4	6.2	23.5	18.4	18.1
	Base-Wiki	2740	33.3	33.7	33.0	11.1	7.7	10.5	4.5	11.6	7.1	31.2	23.4	22.7
	Sim-News	6627	21.8	39.1	39.2	23.2	9.7	22.7	10.0	23.3	9.9	46.6	21.8	23.0
	Sim-Wiki	6174	23.5	40.4	36.1	12.8	6.0	12.7	5.2	13.1	5.3	52.7	31.7	29.7
Test	Translate-Wiki	6189	34.7	31.4	34.0	18.6	9.6	14.2	7.5	16.0	8.8	41.3	20.0	24.6
	MNLI-3k	250	29.2	37.6	33.2	10.6	4.6	9.4	3.7	10.7	4.2	26.3	14.6	15.9
	Base-News	226	38.1	33.2	28.8	12.8	5.7	11.5	5.1	11.6	4.6	22.8	14.4	13.5
	Base-Wiki	234	32.5	32.1	35.5	12.5	8.6	11.7	8.2	11.5	4.8	32.9	24.6	21.1
	Sim-News	219	20.1	44.3	35.6	22.5	11.1	24.9	11.1	23.9	10.9	69.3	20.9	20.6
Test	Sim-Wiki	229	20.5	45.0	34.5	12.6	7.6	13.7	5.8	12.0	4.5	60.5	32.8	28.7
	Translate-Wiki	222	40.5	29.3	30.2	18.7	8.5	13.0	6.9	14.3	6.7	46.3	15.1	21.1

Table 3: Dataset statistics. **HL** denotes the *average* and *standard deviation* of the hypothesis length of each label.

label.

Table 2 shows the agreement statistics for each protocol. BASE shows a higher agreement than SIM and TRANSLATE, although it is lower than MNLI. Compared to MNLI, all of our datasets show higher number of examples with no gold label (no consensus between annotators). As we strictly follow the MNLI protocol for BASE, this suggests that the different population of crowdworkers is likely responsible for these differences.<sup>7</sup>

### 3.1 Dataset Statistics

Table 3 shows the statistics of our collected data. As anticipated, datasets collected using SIM and TRANSLATE have slightly unbalanced distributions compared to BASE. In particular, for SIM, we observe that the entailment class has the lowest distribution in the training and test data.

One clear difference between BASE and our new protocols is the hypothesis length. SIM and TRANSLATE tend to create longer hypothesis than BASE. We suspect that this is an artifact of the sentence-similarity method, which prefers *identical* sentences (both syntax and semantics) over semantically *similar* sentences. Across domains, we observe that sentences from news texts are longer than Wikipedia.

Recent work by McCoy et al. (2019) shows that popular NLI models might learn a simple lexical overlap heuristic for predicting entailment labels. While this heuristic is natural for entailment, it can affect the model’s generalization especially when it is strongly reflected in the data. We calculate word type overlap by using the intersection of premise

<sup>7</sup>MNLI used an organized group of crowdworkers hired through Hybrid (gethybrid.io).

and hypothesis word types, divided by the union of the two sets. The last three columns in Table 3 reports word type overlap in each dataset for each entailment label. We find that word type overlap is a *much* stronger predictor of the label in our new protocols than in BASE. This could be a significant driver of our results and might hurt the generalization performance of models trained using our new protocols’ data.

### 3.2 Annotation Cost

We use the FairWork platform to set payment for each of our HITs (Whiting et al., 2019). FairWork surveys workers to estimate the time that each HIT takes and adjusts pay to a target of US \$15/hr. Based on its estimation, we pay \$0.4 and \$0.3 for each written hypothesis of Base-News and Base-Wiki, respectively. For Sim-News, Sim-Wiki, and Translate-Wiki, we pay \$0.175, \$0.15, \$0.15 for each labeled sentence pair, respectively. In total, we spend \$1791 for each dataset collected from Gigaword and \$1445 for each dataset collected from Wikipedia.

## 4 Experiments

We aim to test whether our alternative protocols can produce high-quality data that yield models that generalize well within NLI and in transfer learning. For the NLI evaluation, we evaluate each model on nine test sets: (i) the five new individual test sets, each containing  $\sim 250$  examples; (ii) the MNLI *development* set; and (iii) the three *development* sets of Adversarial NLI (ANLI; Nie et al., 2020), collected from three rounds of annotation (A1, A2, A3). ANLI is collected using an iterative adversarial approach that follows MNLI but encourages

		Test Data									
Training Data		BN	BW	SN	SW	TW	MNLI	A1	A2	A3	Avg.
<b>CBoW</b>	Base-News	33.4	37.8	32.4	30.1	35.8	35.6	32.8	32.8	<b>33.4</b>	34.0
	Base-Wiki	34.1	33.1	37.9	35.4	39.0	35.6	33.1	31.6	33.2	34.8
	Sim-News	35.4	35.9	32.0	32.3	37.8	35.8	33.1	32.8	<b>33.4</b>	34.3
	Sim-Wiki	32.3	37.2	52.1	49.1	44.6	36.6	33.1	32.4	32.1	38.8
	Translate-Wiki	37.4	39.3	35.4	35.8	45.5	35.4	33.0	<b>32.9</b>	32.8	36.4
<b>RoBERTa</b>	MNLI-3k	79.0	61.3	<b>76.7</b>	57.5	58.1	<b>83.9</b>	33.4	27.0	28.7	56.2
	Base-News	<b>79.4</b>	<b>76.1</b>	57.5	61.6	58.1	83.1	<b>35.8</b>	29.5	28.0	<b>56.6</b>
	Base-Wiki	77.0	74.2	58.5	62.0	61.3	54.0	30.9	31.8	33.1	53.6
	Sim-News	53.3	56.0	65.8	59.8	66.2	79.5	<b>35.8</b>	30.2	28.2	52.8
	Sim-Wiki	62.0	62.8	64.8	<b>64.9</b>	<b>69.1</b>	64.7	32.2	32.0	31.5	53.8
<i>Average per test set</i>		52.0	51.7	52.2	49.7	53.0	54.1	33.2	31.4	31.6	45.4

Table 4: Model performance on individual test sets, as a median over 10 random restarts. **BN**: Base-News, **BW**: Base-Wiki, **SN**: Sim-News, **SW**: Sim-Wiki, **TW**: Translate-Wiki. The last row shows the average performance across models on each test set.

		Test Data									
Training Data		BN	BW	SN	SW	TW	MNLI	A1	A2	A3	Avg.
MNLI-3k		46.5	<b>50.4</b>	33.3	38.4	36.2	<b>52.8</b>	33.3	33.1	33.0	<b>39.7</b>
Base-News		<b>47.8</b>	46.6	33.8	33.6	<b>37.4</b>	51.5	<b>32.5</b>	<b>33.3</b>	33.1	38.8
Base-Wiki		33.2	32.1	<b>44.3</b>	<b>45.0</b>	29.3	32.8	33.3	<b>33.3</b>	33.0	35.1
Sim-News		33.2	35.5	38.8	38.9	29.3	32.8	33.3	<b>33.3</b>	<b>33.5</b>	34.3
Sim-Wiki		33.2	30.8	<b>44.3</b>	44.6	28.8	32.8	33.3	<b>33.3</b>	33.0	34.9
Translate-Wiki		31.4	34.6	34.3	34.5	32.4	33.6	33.3	<b>33.3</b>	<b>33.5</b>	33.4
<i>Average per test set</i>		37.5	38.3	38.1	39.2	32.2	39.4	33.2	33.3	33.2	36.0

Table 5: RoBERTa performance on individual test sets for *hypothesis-only* models.

crowdworkers to write sentences that are difficult for a trained NLI model.

We experiment with two sentence encoders: a CBoW baseline initialized with fastText embeddings (Bojanowski et al., 2017), and a more powerful RoBERTa<sub>Large</sub> (Liu et al., 2019) model, finetuned on individual training sets. We perform a hyperparameter sweep, varying the learning rate  $\in \{1e-3, 1e-4, 1e-5\}$  and the dropout rate  $\in \{0.1, 0.2\}$ . We use batch size of 16 and 4 for CBoW and RoBERTa, respectively. We train each model using the best hyperparameters for 10 epochs, with 10 random restarts. In initial experiments, we find that this setup yields sTable performance given our relatively small datasets, especially when using RoBERTa.<sup>8</sup>

For transfer learning, we test whether each dataset can improve downstream task performance when it is used as intermediate-task data (Phang et al., 2018; Pruksachatkun et al., 2020). As our col-

lected datasets are fairly small (< 10K examples), we use five data-poor downstream target tasks in the SuperGLUE benchmark (Wang et al., 2019a): **COPA** (Roemmele et al., 2011); **WSC** (Levesque et al., 2012); **RTE** (Dagan et al., 2005, et seq), **WiC** (Pilehvar and Camacho-Collados, 2019); and **MultiRC** (Khashabi et al., 2018). We experiment with the BERT<sub>Large</sub> (Devlin et al., 2019) and RoBERTa<sub>Large</sub> models. We follow Pruksachatkun et al. (2020) for training hyperparameters. We use the Adam optimizer (Kingma and Ba, 2015).

We run experiments using the `jiant` toolkit (Wang et al., 2019b), which is the recommended baseline package for SuperGLUE, and is based on Pytorch (Paszke et al., 2019), HuggingFace Transformers (Wolf et al., 2020), and AllenNLP (Gardner et al., 2017).

#### 4.1 NLI Experiments

Table 4 reports the model performance on individual test sets. We include a baseline training data, a 3k randomly sampled training examples from MNLI (MNLI-3k). We observe that all the

<sup>8</sup>This is consistent with the recent findings of Zhang et al. (2020) and Mosbach et al. (2020) regarding fine-tuning BERT-style models on small data.

	Intermediate training data	COPA acc.	MultiRC F1 <sub><math>\alpha</math></sub>	RTE acc.	WiC acc.	WSC acc.	Avg.
BERT	None	70.0	70.9	73.3	72.7	62.5	69.9
	MNLI-3k	+0.0	-0.1	<b>+4.0</b>	-0.8	-2.9	+0.0
	Base-News	<b>+1.0</b>	-0.5	<b>+4.3</b>	-1.7	<b>+1.0</b>	<b>+0.8</b>
	Base-Wiki	<b>+2.0</b>	<b>+0.3</b>	<b>+3.2</b>	-1.2	-1.0	<b>+0.7</b>
	Sim-News	<b>+3.0</b>	-0.3	<b>+2.2</b>	-2.3	+0.0	<b>+0.5</b>
	Sim-Wiki	<b>+7.0</b>	-0.2	<b>+4.0</b>	-2.6	-3.8	<b>+0.9</b>
RoBERTa	Translate-Wiki	<b>+4.0</b>	<b>+0.1</b>	<b>+2.5</b>	-3.7	0.0	<b>+0.6</b>
	None	88.0	77.0	85.2	71.9	67.3	77.9
	MNLI-3k	-4.0	-0.1	+0.7	+0.2	-3.8	-1.5
	Base-News	<b>+1.0</b>	<b>+0.4</b>	<b>+1.1</b>	<b>+0.7</b>	-1.9	<b>+0.3</b>
	Base-Wiki	-2.0	-1.2	<b>+1.1</b>	+0.5	-1.0	-0.5
	Sim-News	-6.0	-3.6	-6.1	-0.1	-3.8	-3.9
	Sim-Wiki	-5.0	-1.9	-2.2	-1.2	-16.3	-5.3
	Translate-Wiki	-5.0	-2.7	-2.5	-1.8	-6.7	-3.7

Table 6: Results on using each collected dataset as intermediate training data on five SuperGLUE tasks. We report the median performance over 3 random restarts on the intermediate NLI models. *None* denotes experiments without intermediate-task training, i.e., direct fine-tuning on target tasks. The last column shows the average score across the five tasks. We report the difference with respect to *None* using BERT and RoBERTa.

CBoW baselines obtain near chance performance. Using RoBERTa, the top performing models are all trained on datasets collected using BASE: Base-News and MNLI-3k. We find that models trained using Translate-Wiki obtain the worst performance. On average across all training sets, ANLI development sets seem to be the hardest, while MNLI seems to be the easiest.

Unsurprisingly, we do not find a single training set which yields the best model across all test sets. We observe that models trained on Base-News perform the best for Base-News and Base-Wiki test sets. Similarly, Sim-Wiki performs the best on both Sim-Wiki and Sim-News test sets. We find that all models do poorly on all ANLI development sets.

Overall, we find that Base-News outperforms all other datasets. However, it is also better than SIM and TRANSLATE which suggests that our new protocols failed. The lower accuracy for SIM and TRANSLATE on their respective test sets also suggests that they produce datasets with noisier labels.

## 4.2 Hypothesis-Only Results

Next, we experiment with a hypothesis-only model (Poliak et al., 2018b) to investigate spurious statistical patterns in the hypotheses which might signal the actual labels to the model. Table 5 reports the results for all five datasets and MNLI. On the five new test sets, we observe that MNLI and Base-News are the most solvable by the hypothesis-only models, though their numbers are still much lower

than with SNLI with accuracy 69.17.

On average across all test sets, none of the training sets obtain much higher performance than chance. All models achieve chance performance on ANLI. However, all of our training sets are fairly small, and these numbers might not be very informative. This also explains why these numbers are relatively lower than other NLI datasets (Poliak et al., 2018b). Across all training sets, we again see that the MNLI test set is the most solvable by the hypothesis-only models.

Our new protocols show lower performance than the BASE, but that may just be because they are of lower overall quality and not because they are less solvable by the hypothesis-only models. We verify this by looking at their transfer learning performance in the following section.

## 4.3 Transfer Learning

Table 6 shows our results when using each collected data as intermediate-training data on the five target tasks. We report the median performance of three random restarts on the validation sets. Using BERT, we observe that all our new datasets yield models with better performance than plain BERT or MNLI-3k as intermediate-training data. We see less positive transfer when we use RoBERTa.

If we look at individual target task performance, both Base-News and Base-Wiki data give consistent positive transfer for RTE, a natural language inference task. We also see some positive trans-

		<b>Entailment</b>	<b>Contradiction</b>	<b>Neutral</b>	
M-3k	looked	0.44	no	1.03	also
	capital	0.43	never	0.95	because
	population	0.43	any	0.88	better
B-News	according	0.58	never	1.07	also
	position	0.45	no	1.02	many
	set	0.42	any	0.90	most
B-Wiki	both	0.45	never	1.18	most
	named	0.38	not	1.01	well
	early	0.35	any	0.96	many
S-Giga	summit	0.53	points	0.66	very
	roads	0.51	we	0.65	research
	weighted	0.46	–	0.59	weeks
S-Wiki	division	0.56	census	0.88	through
	team	0.48	population	0.86	such
	candidate	0.47	2010	0.82	number
T-Wiki	:	0.68	brought	0.45	each
	album	0.58	maintain	0.40	{
	f	0.55	will	0.39	}

Table 7: Top three words most associated with each label by PMI. **M:** MNLI, **B:** Base, **S:** Sim, **T:** Translate.

fer for COPA, however since its validation set is very small (100 examples), we can not conclude anything with confidence.

Overall, our BASE shows better transfer learning performance compared to MNLI, suggesting that our setup is sound. However, we also see that our new protocols perform worse than BASE, showing that they produce less useful training data than the strong baseline of crowdworker writing.

## 5 Dataset Analysis

### 5.1 Annotation Artifacts

Following Gururangan et al. (2018), we compute the PMI between each hypothesis word and label in the training set to examine whether certain words have high associations with its inference label. For a fair comparison, we only use ~3k training examples from each dataset, and sub-sample data collected using SIM and TRANSLATE.

Table 7 shows the top three most associated words for each label, sorted by their PMI scores. We find that BASE has similar associations to MNLI, especially for the neutral and contradiction labels where we found many negations and adverbs. We observe that both SIM and TRANSLATE are less susceptible to this artifact. However, this might be a side-effect of high word overlap in the data, which prefers similar words in the premise and hypothesis. This is also a well-known artifact for NLI data (McCoy et al., 2019).

## 5.2 Qualitative Analysis

Our new protocols use a vector-distance based measurement to find similar sentences, and we find that many of the sentence pairs share similar syntactic structure in their premise and hypothesis, even when both describe different events or entities. We also find that hypothesis in several Sim-News examples differs by only a few words with its premise. For Translate-Wiki, we observe some effects of *translation divergence*, where the translation of the sentence changes semantically because of cross-linguistic distinctions between languages. We provide some examples of these observations in Table 8.

## 6 Related Work

There is a large body of work on constructing data for natural language inference. The first test suite for entailment problems, FraCas (Consortium et al., 1996), is a very small set created manually by experts to isolate phenomena of interest. The RTE challenge corpora (Dagan et al., 2005, et seq) were built by asking human annotators to judge whether a text entails a hypothesis. The SICK dataset (Marelli et al., 2014) is constructed by mining existing paraphrase sentence pairs from image and video captions, which annotators then label.

Some recent works also use automatic methods for generating sentence pairs for entailment data. Zhang et al. (2017) propose a framework to generate hypotheses based on context from general world knowledge or neural sequence-to-sequence methods. The DNC corpus (Poliak et al., 2018a) is an NLI dataset with ordinal judgments constructed by recasting several NLP datasets to NLI examples and labeling them using custom automatic procedures. QA-NLI (Demszky et al., 2018) is an NLI dataset derived from existing QA datasets. Similar to ours, both DNC and QA-NLI use automatic methods to generate sentence pairs. However, neither of them explicitly evaluates whether machine-generated pairs are better than human-generated pairs.

Bowman et al. (2020) propose four potential modifications to the SNLI/MNLI protocol, all still involving crowdworker writing, and show that none yields improvements in the resulting data. SWAG (Zellers et al., 2018) and HellaSwag (Zellers et al., 2019) construct sentence pairs from specific data sources and use language models to generate challenging negative examples.

Type	Dataset	Premise	Hypothesis	Label
Syntactic structure	Sim-News	For many people, choosing wallpaper is one of decorating’s more stressful experiences, fraught with anxiety over color, pattern and cost.	For many people, anxiety about decorating stems from not understanding the language of furniture, fabrics and decorative styles.	E
	Sim-Wiki	Its flowers are pale yellow to white and spherical.	Its flowers are funnel-shaped and pink to white.	C
	Translate-Wiki	But now, in the early 1990s, the Jakarta-Begor railway had turned into a double rail.	However, by the early 1990s, McCreery’s position within the UDA became less secure.	N
Lexical overlap	Sim-News	<b>GrandMet owns Burger King, the world’s second-biggest hamburger chain, as well as US frozen foods manufacturer Pillsbury, which produces the luxury ice-cream Haagen-Daazs.</b>	<b>GrandMet owns Burger King, the world’s second-biggest hamburger chain, as well as US food group Pillsbury, which produces the luxury ice-cream Haagen-Daazs.</b>	E
Translation divergence	Translate-Wiki	Marcus Claudius then abducted her while she was on her way to school.	Marcus Claudius then kidnapped him while he was on his way to school.	N

Table 8: Dataset observations from our new protocols.

On the topic of cost-effective crowdsourcing, Gao et al. (2015) develop a method to reduce redundant translations when collecting human translated data. When the annotation budget is fixed, Khetan et al. (2018) suggest that it is better to label collect single label per training example as many as possible, rather than collecting less training examples with multiple labels.

## 7 Conclusion

In this paper, we introduce two data collection protocols which use fully-automatic pipelines to collect hypotheses, replacing crowdworker writing in the MNLI baseline protocol. We find that switching to a writing-free process with the same source data and annotator pool yields poor-quality data. Our main experiments show strong negative results both in NLI generalization and transfer learning, and mixed results on annotation artifacts, suggesting that MNLI-style crowdworker writing examples are broadly better than automatically paired ones. This finding dovetails with that of Bowman et al. (2020), who find that they are unable to improve upon a base MNLI-style prompt when introducing aids meant to improve annotator speed or creativity. Future work along this line might focus on crowdsourcing strategies (beyond the basic HIT design) which encourage crowdworkers to produce high-quality data with reduced artifacts.

While our fully-automatic methods to construct sentence pairs yield negative results, we have not exhausted all possible automatic techniques for collecting similar sentences. However, given

that we use state-of-the-art tools including FAISS, RoBERTa, and OPUS, and refine our methods with several rounds of piloting and tuning, we are skeptical that there is low-hanging fruit in the two directions we explored. A more radically different direction might involve generating pairs from scratch, using a large language model like GPT-3 (Brown et al., 2020). However, this would still require training data from crowdworker-written dataset, and might add a major source of potentially difficult-to-diagnose bias.

Finally, despite its known issues, we find that MNLI-style data is still the most effective for both NLI evaluation and transfer learning, and future efforts to create similar data should work from that starting point.

## Acknowledgments

This project has benefited from financial support to SB by Eric and Wendy Schmidt (made by recommendation of the Schmidt Futures program), by Samsung Research (under the project *Improving Deep Learning using Latent Structure*), by Intuit, Inc., and in-kind support by the NYU High-Performance Computing Center and by NVIDIA Corporation (with the donation of a Titan V GPU). This material is based upon work supported by the National Science Foundation under Grant No. 1922658. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

## References

- Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A Next-generation Hyperparameter Optimization Framework. In *Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.
- Mikel Artetxe and Holger Schwenk. 2019. Margin-based parallel corpus mining with multilingual sentence embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3197–3203, Florence, Italy. Association for Computational Linguistics.
- Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.
- Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.
- Samuel R. Bowman, Jennimaria Palomaki, Livio Baldini Soares, and Emily Pitler. 2020. Collecting Entailment Data for Pretraining: New Protocols and Negative Results.
- Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners.
- Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.
- The Fracas Consortium, Robin Cooper, Dick Crouch, Jan Van Eijck, Chris Fox, Josef Van Genabith, Jan Jaspars, Hans Kamp, David Milward, Manfred Pinkal, Massimo Poesio, Steve Pulman, Ted Briscoe, Holger Maier, and Karsten Konrad. 1996. Using the Framework.
- Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, pages 177–190. Springer.
- Dorottya Demszky, Kelvin Guu, and Percy Liang. 2018. Transforming Question Answering Datasets Into Natural Language Inference Datasets.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- Mingkun Gao, Wei Xu, and Chris Callison-Burch. 2015. Cost optimization in crowdsourcing translation: Low cost translations made even cheaper. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 705–713, Denver, Colorado. Association for Computational Linguistics.
- Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. AllenNLP: A Deep Semantic Natural Language Processing Platform. Unpublished manuscript available on arXiv.
- Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.
- J. Johnson, M. Douze, and H. Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*.
- Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking Beyond the Surface: A Challenge Set for Reading Comprehension over Multiple Sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 252–262, New Orleans, Louisiana. Association for Computational Linguistics.
- Ashish Khetan, Zachary C. Lipton, and Anima Anandkumar. 2018. Learning From Noisy Singly-labeled Data. In *International Conference on Learning Representations*.
- Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations*,

Hector J. Levesque, Ernest Davis, and Leora Morgenstern. 2012. [The Winograd Schema Challenge](#). In *Proceedings of the Thirteenth International Conference on Principles of Knowledge Representation and Reasoning*, KR’12, pages 552–561. AAAI Press.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. [RoBERTa: A Robustly Optimized BERT Pretraining Approach](#).

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. [A SICK cure for the evaluation of compositional distributional semantic models](#). In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*, pages 216–223, Reykjavik, Iceland. European Languages Resources Association (ELRA).

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. [Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2020. [On the Stability of Fine-tuning BERT: Misconceptions, Explanations, and Strong Baselines](#).

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. [Adversarial NLI: A new benchmark for natural language understanding](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901, Online. Association for Computational Linguistics.

Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2011. English Gigaword Fifth Edition LDC2011T07.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. [PyTorch: An Imperative Style, High-Performance Deep Learning Library](#). In H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Jason Phang, Thibault Févry, and Samuel R. Bowman. 2018. [Sentence Encoders on STILTs: Supplementary Training on Intermediate Labeled-data Tasks](#).

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2019. [WiC: the Word-in-Context Dataset for Evaluating Context-Sensitive Meaning Representations](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1267–1273, Minneapolis, Minnesota. Association for Computational Linguistics.

Adam Poliak, Aparajita Halder, Rachel Rudinger, J. Edward Hu, Ellie Pavlick, Aaron Steven White, and Benjamin Van Durme. 2018a. [Collecting diverse natural language inference problems for sentence representation evaluation](#). In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 67–81, Brussels, Belgium. Association for Computational Linguistics.

Adam Poliak, Jason Naradowsky, Aparajita Halder, Rachel Rudinger, and Benjamin Van Durme. 2018b. [Hypothesis only baselines in natural language inference](#). In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 180–191, New Orleans, Louisiana. Association for Computational Linguistics.

Beatrice Portelli, Jason Zhao, Tal Schuster, Giuseppe Serra, and Enrico Santus. 2020. [Distilling the Evidence to Augment Fact Verification Models](#). In *Proceedings of the Third Workshop on Fact Extraction and VERification (FEVER)*, pages 47–51, Online. Association for Computational Linguistics.

Yada Pruksachatkun, Jason Phang, Haokun Liu, Phu Mon Htut, Xiaoyi Zhang, Richard Yuanzhe Pang, Clara Vania, Katharina Kann, and Samuel R. Bowman. 2020. [Intermediate-task transfer learning with pretrained language models: When and why does it work?](#) In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5231–5247, Online. Association for Computational Linguistics.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. [Know what you don’t know: Unanswerable questions for SQuAD](#). In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S Gordon. 2011. [Choice of Plausible Alternatives: An evaluation of commonsense causal reasoning](#). In *2011 AAAI Spring Symposium Series*.

Holger Schwenk. 2018. [Filtering and mining parallel data in a joint multilingual space](#). In *Proceedings of the 56th Annual Meeting of the Association*

for Computational Linguistics (Volume 2: Short Papers), pages 228–234, Melbourne, Australia. Association for Computational Linguistics.

Holger Schwenk, Vishrav Chaudhary, Shuo Sun, Hongyu Gong, and Francisco Guzmán. 2019. WikiMatrix: Mining 135M Parallel Sentences in 1620 Language Pairs from Wikipedia. *CoRR*, abs/1907.05791.

Jörg Tiedemann and Santhosh Thottingal. 2020. OPUS-MT — Building open translation services for the World. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation (EAMT)*, Lisbon, Portugal.

Harsh Trivedi, Heeyoung Kwon, Tushar Khot, Ashish Sabharwal, and Niranjan Balasubramanian. 2019. Repurposing entailment for multi-hop question answering tasks. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2948–2958, Minneapolis, Minnesota. Association for Computational Linguistics.

Masatoshi Tsuchiya. 2018. Performance impact caused by hidden bias of training data for recognizing textual entailment. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*, Miyazaki, Japan. European Languages Resources Association (ELRA).

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019a. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*.

Alex Wang, Ian F. Tenney, Yada Pruksachatkun, Phil Yeres, Jason Phang, Haokun Liu, Phu Mon Htut, Katherin Yu, Jan Hula, Patrick Xia, Raghu Pappagari, Shuning Jin, R. Thomas McCoy, Roma Patel, Yinghui Huang, Edouard Grave, Najoung Kim, Thibault Févry, Berlin Chen, Nikita Nangia, Anhad Mohananey, Katharina Kann, Shikha Bordia, Nicolas Patry, David Benton, Ellie Pavlick, and Samuel R. Bowman. 2019b. jiant 1.3: A software toolkit for research on general-purpose text understanding models. <http://jiant.info/>.

Mark E Whiting, Grant Hugh, and Michael S Bernstein. 2019. Fair Work: Crowd Work Minimum Wage with One Line of Code. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, pages 197–206.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume*

1 (Long Papers)

, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierrette Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Huggingface’s transformers: State-of-the-art natural language processing.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. SWAG: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 93–104, Brussels, Belgium. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Sheng Zhang, Rachel Rudinger, Kevin Duh, and Benjamin Van Durme. 2017. Ordinal common-sense inference. *Transactions of the Association for Computational Linguistics*, 5:379–395.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q. Weinberger, and Yoav Artzi. 2020. Revisiting Few-sample BERT Fine-tuning.

## A Appendices

### A.1 Indexing and Retrieval

**Gigaword** The corpus contains texts from seven news sources: afp\_eng, apw\_eng, cna\_eng, ltw\_eng, nyt\_eng, wpb\_eng, and xin\_eng. We build one index for each news source with type “PCAR64, IVFx, Flat”, where  $x$  defines the number of clusters in the index. This type of index allows faster retrieval, however it requires a training stage to assign a centroid to each cluster. We refer readers to FAISS documentation for more detail explanations.<sup>9</sup>

For each news source, we randomly sample 100 sentences from its monthly articles and use them as seed sentences to train the clusters. We then set the number of clusters  $x$  to  $\frac{N}{100}$  (rounded to the nearest hundred), where  $N$  is the number of seed sentences. Table 9 lists the number of seed sentences and clusters used for each news source index.

Source	#seed sentences	x
afp_eng	111,147	1,100
apw_eng	146,119	1,400
cna_eng	125,508	1,200
ltw_eng	90,195	900
nyt_eng	136,827	1,300
wpb_eng	9,144	100
xin_eng	157,760	1,500

Table 9: Number of seed sentences and number of clusters for each news source index.

During retrieval, for each query, we retrieve top 1000 sentences from each index and perform reranking on the combined list, i.e., 7,000 sentence pairs, as described in Section 2.2.

**Wikipedia** We build one index for the whole Wikipedia corpus. For seed sentences, we use sentences taken from the first paragraph of each article as it usually contains the summary of the article. We set the number of clusters  $x$  to 15,000.

<sup>9</sup><https://github.com/facebookresearch/faiss>

## A.2 Writing HIT Instructions

Instructions
<p>The New York University Center for Data Science is collecting your answers for use in research on computer understanding of English. Thank you for your help!</p> <p>This task will involve reading a prompt and writing three sentences that relate to it. The prompt will describe a situation or event. Using only this description and what you know about the world, please:</p> <ul style="list-style-type: none"><li>• Write one sentence that is <b>definitely correct</b> about the situation or event in the prompt.</li><li>• Write one sentence that <b>maybe correct</b> about the situation or event in the prompt.</li><li>• Write one sentence that is <b>definitely incorrect</b> about the situation or event in the prompt.</li></ul> <p>These prompts were taken from news and Wikipedia articles. If you recognize an individual, place, or event, please do not use outside knowledge about it to write your sentences. Please write sentences based only on the prompt and general beliefs or assumptions about what happens in the world.</p> <p>If you have more questions, please consult our FAQ.</p> <p><b>Prompt:</b> <i>"Security and reliability are two important aspects of this service because of the sensitivity and urgency of the data sent over."</i></p> <p><b>Definitely correct</b> Example: For the prompt "<i>The cottages near the shoreline, styled like plantation homes with large covered porches, are luxurious within; some come with private hot tubs.</i>", you could write "<i>The shoreline has plantation style homes near it, which are luxurious and often have covered porches or hot tubs.</i>"</p> <div style="border: 1px solid black; height: 1.2em; width: 100%;"></div> <p><b>Maybe correct</b> Example: For the prompt "<i>Government Executive magazine annually presents Government Technology Leadership Awards to recognize federal agencies and state governments for their excellent performance with information technology programs.</i>", you could write "<i>In addition to their annual Government Technology Leadership Award, Government Executive magazine also presents a cash prize for best dressed agent from a federal agency.</i>"</p> <div style="border: 1px solid black; height: 1.2em; width: 100%;"></div> <p><b>Definitely incorrect</b> Example: For the prompt "<i>Yes, he's still under arrest, which is why USAT's front-page refer headline British Court Frees Chile's Pinochet is a bit off.</i>", you could write "<i>The headline 'British Court Frees Chile's Pinochet' is correct, since the man is freely roaming the streets.</i>"</p> <div style="border: 1px solid black; height: 1.2em; width: 100%;"></div> <p><b>Problems (optional)</b> If something is wrong with the prompt that makes it difficult to understand, let us know here.</p> <div style="border: 1px solid black; height: 1.2em; width: 100%;"></div>

Figure 2: Writing HIT instructions.

### A.3 Data Labeling and Validation HIT Instructions

Instructions
<p>The New York University Center for Data Science is collecting your answers for use in research on computer understanding of English. Thank you for your help!</p> <p>Your job is to figure out, based on a correct prompt (<b>S1</b>), if another prompt (<b>S2</b>) is also correct:</p> <ul style="list-style-type: none"><li>Choose <b>definitely correct</b> if any event or situation that can be described by S1 would also fit S2. Example: S1: "A kitten with spots is playing with yarn." S2: "A cat is playing"</li><li>Choose <b>maybe correct</b> if S2 could describe an event or situation that fit S1, but could also describe sentences that don't fit S1. Example: S1: "A kitten with spots is playing with yarn." S2: "A kitten is playing with yarn on a sofa"</li><li>Choose <b>definitely incorrect</b> if any event or situation that could possibly described with S1 would not fit S2. Example: S1: "A kitten with spots is playing with yarn." S2: "A puppy is playing with yarn."</li><li>Choose <b>I don't understand</b> if there is something wrong with the prompts that make them hard to understand (more than just a typo). Example: S1: "A kitten with spots is playing with yarn." S2: "Marie talks (Oct. 26 - Nov."</li></ul> <p>More questions? Visit our <a href="#">FAQ</a>.</p> <p><b>Question 1</b> S1: \${premise} S2: \${hypothesis}</p> <p><input type="radio"/> definitely correct    <input type="radio"/> maybe correct    <input type="radio"/> definitely incorrect    <input type="radio"/> I don't understand</p> <p><b>Question 2</b> S1: \${premise} S2: \${hypothesis}</p> <p><input type="radio"/> definitely correct    <input type="radio"/> maybe correct    <input type="radio"/> definitely incorrect    <input type="radio"/> I don't understand</p> <p>...</p> <p><b>Question 5</b> S1: \${premise} S2: \${hypothesis}</p> <p><input type="radio"/> definitely correct    <input type="radio"/> maybe correct    <input type="radio"/> definitely incorrect    <input type="radio"/> I don't understand</p>

Figure 3: Data Labeling and Validation HIT instructions. We collect one annotation per example for data labeling and five annotations per example for validation.

# MaP: A Matrix-based Prediction Approach to Improve Span Extraction in Machine Reading Comprehension

Huaishao Luo<sup>1\*</sup>, Yu Shi<sup>2</sup>, Ming Gong<sup>3</sup>, Linjun Shou<sup>3</sup>, Tianrui Li<sup>1</sup>

<sup>1</sup>School of Information Science and Technology, Southwest Jiaotong University

<sup>2</sup>Microsoft Cognitive Services Research Group

<sup>3</sup>Microsoft STCA NLP Group

huaishaoluo@gmail.com, trli@sjtu.edu.cn

{yushi,migon,lisho}@microsoft.com

## Abstract

Span extraction is an essential problem in machine reading comprehension. Most of the existing algorithms predict the start and end positions of an answer span in the given corresponding context by generating two probability vectors. In this paper, we propose a novel approach that extends the probability vector to a probability matrix. Such a matrix can cover more start-end position pairs. Precisely, to each possible start index, the method always generates an end probability vector. Besides, we propose a sampling-based training strategy to address the computational cost and memory issue in the matrix training phase. We evaluate our method on SQuAD 1.1 and three other question answering benchmarks. Leveraging the most competitive models BERT and BiDAF as the backbone, our proposed approach can get consistent improvements in all datasets, demonstrating the effectiveness of the proposed method.

## 1 Introduction

Machine reading comprehension (MRC), which requires the machine to answer comprehension questions based on the given passage of text, has been studied extensively in the past decades (Liu et al., 2019). Due to the increase of various large-scale datasets (e.g., SQuAD (Rajpurkar et al., 2016) and MS MARCO (Nguyen et al., 2016)), and the enhancement of pre-trained models (e.g., ELMo (Peters et al., 2018), BERT (Devlin et al., 2019), and XLNet (Yang et al., 2019)), remarkable advancements have been made recently in this area. Among various MRC tasks, span extraction is one of the essential tasks. Given the context and question, the span extraction task is to extract a span of the most plausible text from the corresponding context as a

\*This work was done during the first author’s internship at Microsoft

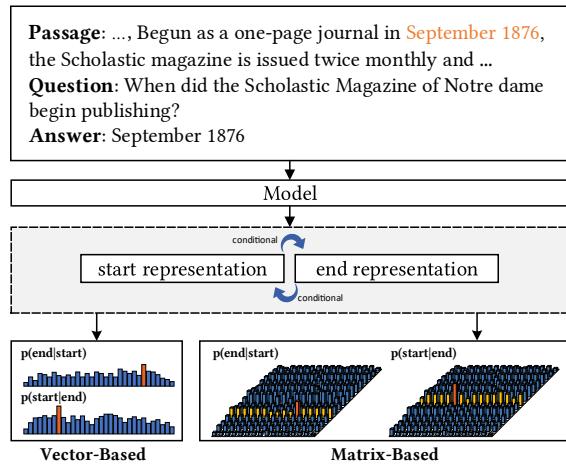


Figure 1: An illustration of a machine reading comprehension framework. Most of previous works are vector-based approaches shown as the left part. Our matrix-based conditional approach is shown in the right part. In our setting, every start (or end) position has an end (or start) probability vector, which leads that the output probabilities is a matrix (best seen in color).

candidate answer. Although there exist unanswerable cases beyond the span extraction, the span-based task is still fundamental and significant in the MRC field.

Previous methods used to predict the start and end position of an answer span can be divided into two categories. The first one regards the generation of begin position and end position independently. We refer to this category as *independent approach*. It can be written as  $p_* = p(*|\mathbf{H}^*)$ , where  $*$   $\in \{s, e\}$ , the  $s$  and  $e$  denote start and end, respectively.  $\mathbf{H}^*$  is the hidden representation, in which  $\mathbf{H}^s$  and  $\mathbf{H}^e$  usually have shared features. The other one constructs a dependent route from the start position when predicting the end position. We refer to this category as *conditional approach*. It can be formalized as  $p_s = p(s|\mathbf{H}^s), p_e = p(e|s, \mathbf{H}^e)$ . This category usu-

ally reuses the predicted position information (e.g.,  $s$ ) to assist in the subsequent prediction. The difference between these two approaches is that the conditional approach considers the relationship between start and end positions, but the independent approach does not. In the literature, AMANDA (Kundu and Ng, 2018b), QANet (Yu et al., 2018), and SEBert (Keskar et al., 2019) can be regarded as the independent approach, where the probabilities of the start and end positions are calculated separately with different representations. DCN (Xiong et al., 2017), R-NET (Wang et al., 2017), BiDAF<sup>1</sup> (Seo et al., 2017), Match-LSTM (Wang and Jiang, 2017), S-Net (Tan et al., 2018), SDNet (Zhu et al., 2018), and HAS-QA (Pang et al., 2019) belong to the conditional approach. The probabilities are generated in sequence.

The conditional approach empirically has an advantage over the independent approach. However, the output distributions of the previous conditional approaches are two probability vectors. It ignores some more possible start-end pairs. As an extension, every possible start (or end) position should have an end (or start) probability vector. Thus, the output conditional probabilities is a matrix.

We propose a **M**atrix-based **P**rediction approach (**MaP**) based on the above consideration in this paper. As Figure 1 shown, the key point is to consider as many probabilities as possible in *training* and *inference* phases. Specifically, we calculate a conditional probability matrix instead of a probability vector to expand the choices of start-end pairs. Because of more values contained in a matrix than a vector, there is a big challenge in the training phase of the MaP. That is the high computational cost and memory issues if the input sequence is long. As an instance, the matrix contains 262,144 probability values if the sequence length is 512. Therefore, we propose a sampling-based training strategy to speed up the training and reduce the memory cost.

The main contributions of our work are four-fold.

- A novel conditional approach is proposed to address the limitation of the probability vector generated by the vector-based conditional approach. It increases the likelihood of hitting the ground-truth start and end positions.
- A sampling-based training strategy is pro-

<sup>1</sup>We classify BiDAF as a conditional approach by its official implementation: <https://github.com/allenai/bi-att-flow>

posed to overcome the computation and memory issues in the training phase of the matrix-based conditional approach.

- An ensemble approach on both start-to-end and end-to-start directions of conditional probability is investigated to improve the accuracy of the answer span.
- We evaluate our strategy on SQuAD 1.1 and three other question answering benchmarks. The implementation of the matrix-based conditional approach is designed based on the BERT and BiDAF, which are the most competitive models, to test the generalization of our strategy. The consistent improvements in all datasets demonstrate the effectiveness of the strategy.

## 2 Methodology

In this section, we first give the problem definition. Then we introduce a typical vector-based conditional approach. Next, we mainly introduce our matrix-based conditional approach and sampling-based training strategy. Finally, an ensemble approach on both start-to-end and end-to-start directions of conditional probability is discussed.

### 2.1 Problem Statement

Given the passage  $P = \{t_1, t_2, \dots, t_n\}$  and the question  $Q = \{q_1, q_2, \dots, q_m\}$ , the span extraction task needs to extract the continuous subsequence  $A = \{t_s, \dots, t_e\}$  ( $1 \leq s \leq e \leq n$ ) from the passage as the right answer to the question, where  $n$  and  $m$  are the length of the passage and question respectively,  $s$  and  $e$  are the start and end position in the passage. Usually, the objective to predict  $a = (s, e)$  is maximizing the conditional probability  $p(a|P, Q)$ .

### 2.2 A Typical Vector-based Approach

We summarize a typical implementation of the vector-based conditional approach shown in Figure 2. Previous mentioned R-NET, BiDAF, Match-LSTM, S-Net, and SDNet can be regarded as such implementation. Its backbone is the Pointer Network proposed by Vinyals et al. (2015). The interactive representation  $\mathbf{H} \in \mathbb{R}^{n \times d}$  between the given question  $Q$  and passage  $P$  is calculated as follows,

$$\mathbf{H} = \mathfrak{M}(Q, P), \quad (1)$$

where  $\mathfrak{M}$  is a neural network, e.g., Match-LSTM, QANet, BERT, and XLNet,  $d$  is the dimension size

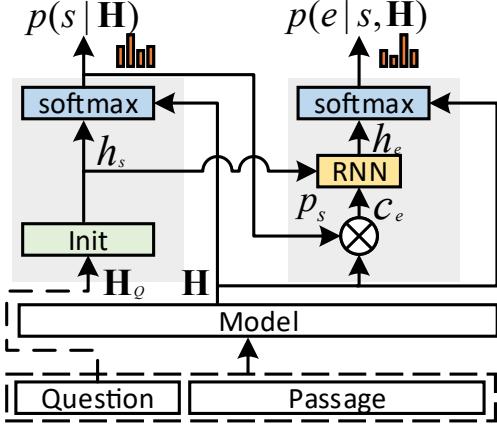


Figure 2: A typical implementation of the vector-based conditional approach.

of the representation. After generating the interactive representation, the next step is to predict the answer span.

The main architecture of the span prediction is an RNN. As an instance, LSTM is used in (Wang and Jiang, 2017), and GRU is adopted in (Tan et al., 2018; Zhu et al., 2018). Take the hidden representation  $h_e \in \mathbb{R}^k$  of end position as an example, which is calculated as follows,

$$h_e = \text{RNN}(h_s, c_e), \quad (2)$$

$$c_e = \mathbf{H}^\top p_s, \quad (3)$$

where  $p_s = p(s|\mathbf{H})$  is the start probability and  $p_s \in \mathbb{R}^n$ ,  $k$  is the dimension size of  $h_e$ . Then  $p_e = p(e|s, \mathbf{H}) (p_e \in \mathbb{R}^n)$  can be calculated using  $h_e$  as follows,

$$p(e|s, \mathbf{H}) = \text{softmax}\left(\mathbf{v}^\top \tanh\left(\mathbf{V}\mathbf{H}^\top + [\mathbf{W}_e h_e]^n\right)\right) \quad (4)$$

where  $[\cdot]^n$  is an operation that generates a matrix by repeating the vector on the left  $n$  times,  $\mathbf{v} \in \mathbb{R}^l$ ,  $\mathbf{V} \in \mathbb{R}^{l \times d}$ , and  $\mathbf{W}_e \in \mathbb{R}^{l \times k}$  are parameters to be learned.

The calculation of  $p(s|\mathbf{H})$  is similar to  $p(e|s, \mathbf{H})$ . The key is to obtain the hidden state  $h_s$ . A choice is to use an attention approach to condense the question representation into a vector. The process is as follows,

$$p_{init} = \text{softmax}\left(\mathbf{v}_Q^\top \tanh\left(\mathbf{V}_Q \mathbf{H}_Q^\top\right)\right), \quad (5)$$

$$h_s = \mathbf{H}_Q^\top p_{init}, \quad (6)$$

where  $\mathbf{H}_Q \in \mathbb{R}^{m \times d}$  is the representation corresponding to  $Q$ ,  $\mathbf{v}_Q \in \mathbb{R}^l$ , and  $\mathbf{V}_Q \in \mathbb{R}^{l \times d}$  are parameters.

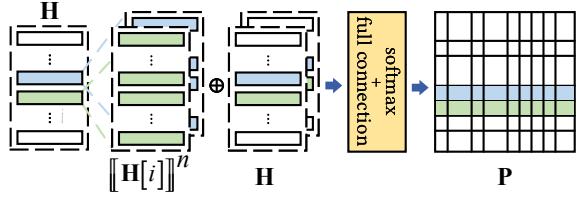


Figure 3: Matrix-based conditional approach.

There is a vast number of works on MRC. However, most of these works focus on the design of  $\mathfrak{M}$  and generate the answer span based on the vector-based conditional approach. In this paper, we expand the vector to a probability matrix. Thus, many more possibilities can be covered. It is also a natural manner because that every start (or end) position should have an end (or start) probability vector.

### 2.3 Matrix-based Conditional approach

As the previous description, the implementation of the vector-based conditional approach has a unified and important implementation step: create a ‘condition’. Take the forward direction (‘condition’ constructed from the start position to end position) of the vector-based conditional approach as an example, the ‘condition’ is the probability vector  $p_s$ . The end probability vector  $p_e$  can not be calculated until generating  $p_s$ . However, there is only one probability vector  $p_e$  whatever the start position is. In this paper, we keep the ‘condition’ step but propose calculating an individual  $p_e$  for each start position. Specifically, the probability matrix  $\mathbf{P}_e \in \mathbb{R}^{n \times n}$  is calculated as follows,

$$\mathbf{P}_e^{(i)} = \text{softmax}\left(\mathbf{v}^\top \tanh\left(\mathbf{V}\left[\mathbf{H}^\top; [\mathbf{H}[i]]^n\right]\right)\right) \quad (7)$$

where  $\mathbf{P}_e^{(i)}$  denotes the  $i$ -th row of  $\mathbf{P}_e$ ,  $[\cdot]$  is a concatenate operation,  $[\cdot]^n$  is an operation that generates a matrix by repeating the vector on the left  $n$  times,  $[i]$  means to choose the  $i$ -th row from the matrix  $\mathbf{H}$ ,  $\mathbf{v} \in \mathbb{R}^l$  and  $\mathbf{V} \in \mathbb{R}^{l \times 2d}$  are parameters. Figure 3 illustrates the calculation process of Eq. (7).

Although the calculation is brief and can cover more probabilities than the vector-based approach, there is a big question on computation cost and memory occupation. The main computation cost comes from the matrix multiplication between  $\mathbf{V}$  and  $\left[\mathbf{H}^\top; [\mathbf{H}[i]]^n\right]$  in Eq. (7), totally  $n$  times

such computation for  $\mathbf{P}_e$ . The number of probabilities is also  $n$  times bigger than the vector-based conditional approach. It also causes the issue of out of memory (OOM), especially with a big  $n$ , due to intermediate gradient values needing cache in the training phase. We propose a sampling-based training strategy to solve the above issues.

## 2.4 Sampling-based Training Strategy

In order to train the probability matrix effectively, we propose a sampling-based strategy in the training phase. Given the hyper-parameter  $k$ , we first choose the indexes  $\hat{\mathcal{I}}$  of top  $k - 1$  possibilities from  $p_s^{(-\hat{s})}$ ,

$$\hat{\mathcal{I}} = \text{top}\left(p_s^{(-\hat{s})}, k - 1\right), \quad (8)$$

where  $\text{top}(p, v)$  is an operation used to get the indexes of top  $v$  values in  $p$ ,  $p^{(-w)}$  contains all but  $w$ -th value of  $p$ , and  $\hat{s}$  is the truth start position used as the supervised information in the training phase. Then, the  $\hat{s}$  must merge to  $\hat{\mathcal{I}}$ ,

$$\mathcal{I} = \hat{\mathcal{I}} + \{\hat{s}\}, \quad (9)$$

where  $\mathcal{I}$  contains  $k$  indexes.

Eq. (8) and Eq. (9) promise that the sampled start probabilities must contain and only contain the target probability which we need to train in each iteration. The target probability is the  $\hat{s}$ -th value in  $p_s$ , and the bigger, the better.

After sampling the start probability vector, the computation cost of  $\mathbf{P}_e$  decrease. For each  $i \in \mathcal{I}$ , executing Eq. (7) repeatedly can generate a sampling-based end probability matrix. It is noted that this sampling-based matrix is a part of the original  $\mathbf{P}_e$ . We refer to it as  $\tilde{\mathbf{P}}_e$ , and  $\tilde{\mathbf{P}}_e \in \mathbb{R}^{k \times n}$ . It is still a big issue of computation cost and memory occupation for  $\tilde{\mathbf{P}}_e$  with a long sequence. So, we carry out similar operations in Eq. (8) and Eq. (9) for each row of  $\tilde{\mathbf{P}}_e$  using  $\hat{e}$  instead of  $\hat{s}$ , where  $\hat{e}$  is the end truth position. Finally, the sampling-based matrix  $\hat{\mathbf{P}}_e \in \mathbb{R}^{k \times k}$  is generated. It is small enough to train compared with  $\mathbf{P}_e$ . Figure 4 shows the sampling results colored with a yellow background on the left and corresponding ground truth matrix on the right.

## 2.5 Training

In the training phase, the objective function is to minimize the cross-entropy error averaged over start and end positions,

$$\mathcal{L} = \frac{1}{2}(\mathcal{L}_s + \mathcal{L}_e), \quad (10)$$

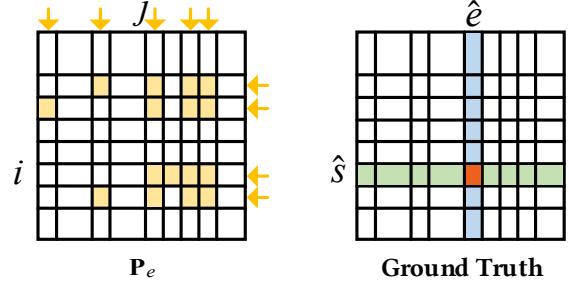


Figure 4: A sampling of probability matrix. Left: the calculated probability matrix with sampled top four positions (in both row and column directions colored with yellow background). Right: the ground truth matrix, where position  $(\hat{s}, \hat{e})$  with the red background has probability 1.

$$\mathcal{L}_s = -\frac{1}{N} \sum_{i=1}^N \left( \mathbb{I}(\hat{s}) (\log(p_s))^\top \right), \quad (11)$$

$$\mathcal{L}_e = -\frac{1}{N} \sum_{i=1}^N \left( \mathcal{T}(\mathbb{I}(\hat{s}, \hat{e})) (\log(\mathcal{T}(\hat{\mathbf{P}}_e)))^\top \right), \quad (12)$$

where  $N$  is the number of data,  $\mathbb{I}(\hat{s})$  means the one-hot vector of  $\hat{s}$ ,  $\mathbb{I}(\hat{s}, \hat{e})$  means a zero matrix with a value of 1 in row  $\hat{s}$  and column  $\hat{e}$ , and  $\mathcal{T}()$  is a row wise flatten operation. The flatten operation makes the loss function on matrix-based distribution similar to that on vector-based distribution.

As the introduction of the sampling-based training strategy, there are limited end probabilities that could be trained in each iteration. The extreme situation is  $k$  equals to  $n$ , which makes all probability matrix calculate each time. As our previous argumentation, it is almost impossible for time and memory limitations. However, there is a question of what makes sampling strategy works. The following content gives some explanation based on gradient backpropagation.

The gradient of the cross-entropy  $\mathcal{L}_*$  to the predicted logits  $z_*$  is,

$$\frac{\partial \mathcal{L}_*}{\partial z_*} = \begin{cases} p_*^{(i)} - 1, & \text{if } i \text{ is the ground-truth;} \\ p_*^{(j)}, & \text{others} \end{cases} \quad (13)$$

where  $p_* = \text{softmax}(z_*)$  is probabilities in which values are between 0 and 1 (exclusion). Thus  $p_*^{(i)} - 1$  is negative, and  $p_*^{(j)}$  is positive in most cases. As the parameters  $\theta$  update usually follows  $\theta_t = \theta_{t-1} - \eta \cdot \nabla_\theta \mathcal{L}(\theta)$  and learning rate  $\eta$  is a positive value, the probability in ground-truth position should go up, and the probabilities in other sampled positions should go down.

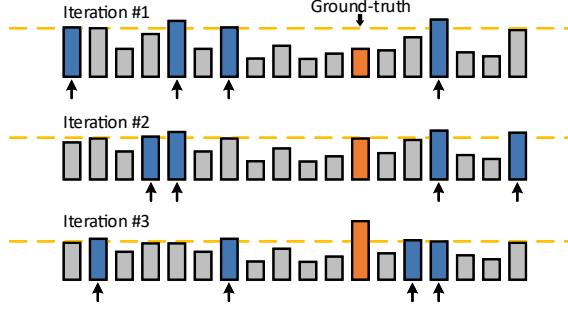


Figure 5: Sampling-based probabilities training ( $k = 5$ ). Block with red color is the ground-truth, blocks with blue color are the sampled probabilities. Probabilities with a gray background will not change their values in each iteration.

Figure 5 illustrates the sampling-based training process, where the parameter  $k$  is set to 5. It means that there are extra top-4 probabilities (blue background) except ground-truth (red background) will be chosen to calculate. With the iteration going from #1 to #3, the probability in ground-truth position goes up, and that in sampled top-4 positions goes down. Such a sampling-based training approach has the same goal with the training on the whole probabilities, thus should have proximity results.

## 2.6 Ensemble for Inference

The vector-based conditional approach usually searches the span  $(s, e)$  via the computation of  $p_s^{(i)} \times p_e^{(j)}$  under the condition of  $i \leq j$ , and choices the  $(i^*, j^*)$  with the highest  $p_s^{(i^*)} \times p_e^{(j^*)}$  as the output in the inference phase. The matrix-based conditional approach follows the same idea, but the calculation of the probability is  $p_s^{(i)} \times \mathbf{P}_e^{(i,j)}$  instead of  $p_s^{(i)} \times p_e^{(j)}$ . The  $p_s^{(i)}$  is the  $i$ -th probability in  $p_s$ , and  $\mathbf{P}_e^{(i,j)}$  is the probability in row  $i$ , column  $j$  of  $\mathbf{P}_e$ .

The above inference strategy only involves one direction, e.g., start-to-end direction (generate start position firstly, then generate end position), which is the most cases in previous works. An ensemble of both start-to-end and end-to-start directions is a good choice to improve the performance. The difference in end-to-start direction is that Eqs. (7-12) should be repeated in the opposite direction. In other words, the start is replaced by  $e$ , and the end is replaced by  $s$ . Totally, there are two groups of probabilities,  $(p_s, \mathbf{P}_e)$  and  $(p_e, p_s)$ . In this paper, we design a type of ensemble strategy, which first chooses top  $k$  pairs  $\mathbf{F} = \{(i_f, j_f)\}$  with

---

### Algorithm 1 MaP Training Algorithm

---

**Input:**  $N$  pairs of passage  $P$  and question  $Q$ ,  $k$  used to choose top probabilities;

**Output:** Learned MaP model

- 1: Initialize all learnable parameters  $\Theta$ ;
- 2: **repeat**
- 3:   Select a batch of pairs from corpus;
- 4:   **for** each pair  $(P, Q)$  **do**
- 5:     Use a neural network  $\mathfrak{M}$  to generate the representation  $\mathbf{H}$ ; (Eq. 1)
- 6:     Compute start probability vector  $p_s$ ; (Eqs. 4-6)
- 7:     Sample indexes  $\mathcal{I}$  by choosing top  $k - 1$  probabilities of  $p_s$ ; (Eqs. 8,9)
- 8:     Compute end probability matrix  $\mathbf{P}_e$ ; (Eq. 7)
- 9:     Compute objective  $\mathcal{L}$ ; (Eq. 10-12)
- 10:   **end for**
- 11:   Use the backpropagation algorithm to update parameters  $\Theta$  by minimizing the objective with the batch update mode
- 12: **until** stopping criteria is met

---

highest probability  $p_s^{(i_f)} \times \mathbf{P}_e^{(i_f, j_f)}$ , then chooses top  $k$  pairs  $\mathbf{B} = \{(j_b, i_b)\}$  with highest probability  $p_e^{(j_b)} \times \mathbf{P}_s^{(j_b, i_b)}$ . It is noted that some pairs may have the same position, e.g.,  $(3_f, 5_f)$  and  $(5_b, 3_b)$ . If there are the same elements, we prune away them in  $\mathbf{B}$ . Then, we choose the  $(i^*, j^*)$  with highest probability in  $\mathbf{F} \cup \mathbf{B}$ .

The overall training procedure of MaP is summarized in Algorithm 1.

## 3 Experiments

In this section, we conduct experiments to evaluate the effectiveness of the proposed MaP.

### 3.1 Datasets

We first evaluate our strategy on SQuAD 1.1, which is a reading comprehension benchmark. The benchmark benefits to our evaluation compared with its augmented version SQuAD 2.0 due to its questions always have a corresponding answer in the given passages. We also evaluate our strategy on three other datasets from the MRQA 2019 Shared Task<sup>2</sup>: NewsQA (Trischler et al., 2017), HotpotQA (Yang et al., 2018), Natural Questions (Kwiatkowski et al., 2019). As the SQuAD 1.1 dataset, the format of

<sup>2</sup><https://github.com/mrqa/MRQA-Shared-Task-2019>

Models		SQuAD		NewsQA		HotpotQA		Natural Questions	
		EM	F1	EM	F1	EM	F1	EM	F1
BERT-Base	InD	81.24	88.38	52.59	67.12	59.01	75.69	67.31	78.96
	<b>MaP<sub>F</sub></b>	81.78	88.59	52.66	66.50	59.82	75.81	67.68	78.99
	<b>MaP<sub>E</sub></b>	<b>82.12</b>	<b>88.63</b>	<b>53.06</b>	<b>67.37</b>	<b>60.55</b>	<b>76.12</b>	<b>68.21</b>	<b>79.09</b>
BERT-Large	InD	84.05	90.85	54.46	69.61	62.26	78.18	69.44	80.93
	<b>MaP<sub>F</sub></b>	84.50	90.89	54.84	68.73	63.19	78.99	69.56	80.49
	<b>MaP<sub>E</sub></b>	<b>84.79</b>	<b>90.89</b>	<b>55.29</b>	<b>69.98</b>	<b>63.70</b>	<b>79.25</b>	<b>69.91</b>	<b>81.22</b>
BiDAF	VCP	68.57	78.23	44.04	58.07	47.31	62.42	56.95	68.79
	<b>MaP<sub>F</sub></b>	68.85	78.06	44.19	58.65	50.25	65.21	57.04	68.87
	<b>MaP<sub>E</sub></b>	<b>69.55</b>	<b>78.91</b>	<b>44.25</b>	<b>58.91</b>	<b>51.45</b>	<b>66.74</b>	<b>57.21</b>	<b>69.08</b>

Table 1: The performance (%) of EM and F1 on SQuAD 1.1 and three MRQA extractive question answering tasks.  $\text{MaP}_F$  is the matrix-based conditional approach calculating on start-to-end direction.  $\text{MaP}_E$  means the ensemble of both directions of matrix-based conditional approach. InD denotes the independent approach. VCP is vector-based conditional approach.

the task is extractive question answering. It contains no unanswerable or non-span answer questions. Besides, the fact that these datasets vary in both domain and collection pattern benefits for the evaluation of our strategy on generalization across different data distributions. Table 2 shows the statistics of these datasets.

Dataset	Training	Development
SQuAD 1.1	86,588	10,507
NewsQA	74,160	4,212
HotpotQA	72,928	5,904
Natural Questions	104,071	12,836

Table 2: The statistics of datasets.

### 3.2 Baselines

To validate the effectiveness and generalization of our proposed strategy on the span extraction, we implement it using two strong backbones, BERT and BiDAF. Specifically, we borrow their main bodies except the top layer to implement the proposed strategy to finish the span extraction on different datasets. Some more tests on other models, e.g., XLNet (Yang et al., 2019) and SpanBERT (Joshi et al., 2019), and datasets will be our future work.

- **BERT:** BERT is an empirically powerful language model, which obtained state-of-the-art results on eleven natural language processing tasks in the past (Devlin et al., 2019). The original implementation in their paper on the span prediction task belongs to the independent approach. Both BERT-base and BERT-large with

uncased pre-trained weights are used in comparison to investigating the effect of the ability of language model on span extraction with different prediction approaches.

- **BiDAF:** BiDAF is used as a baseline of the vector-based conditional approach (Seo et al., 2017). The use of a multi-stage hierarchical process and a bidirectional attention flow mechanism makes its representation powerful.

There are four strategies of span extraction involved in our comparison: **InD** denotes the independent approach; **VCP** is the vector-based conditional approach; **MaP<sub>F</sub>** is our matrix-based conditional approach calculating on start-to-end direction; **MaP<sub>E</sub>** means the ensemble of both directions of matrix-based conditional approach. The InD is used to compare with MaP<sub>F</sub> and MaP<sub>E</sub> in BERT, and the VCP is used to compare with MaP<sub>F</sub> and MaP<sub>E</sub> in BiDAF.

### 3.3 Experimental Settings

We implement the BERT and BiDAF following the official settings for a fair comparison. For the BERT, we train for 3 epochs with a learning rate of  $5e-5$  and a batch size of 32. The max sequence length is 384 for SQuAD 1.1 and 512 for other datasets, and a sliding window of size 128 is used for all datasets if the sentence is longer than the max length. For the BiDAF, we keep all original settings except a difference that we use ADAM (Kingma and Ba, 2015) optimizer with a learning rate of  $1e-3$  in the training phase instead of AdaDelta (Zeiler, 2012) for a stable performance.

Following the work from (Rajpurkar et al., 2016), we evaluate the results using Exact Match (EM) and Macro-averaged F1 score. The sampling parameter  $k$  is set to 20 for our strategy. We implement our model in python using the pytorch-transformers library<sup>3</sup> for BERT and the AllenNLP library<sup>4</sup> for BiDAF. The reported results are average scores of 5 runs with different random seeds. All computations are done on 4 NVIDIA Tesla V100 GPUs.

### 3.4 Main Results

The results of our strategies as well as the baselines are shown in Table 1. All these values come from the evaluation of the development sets in each dataset due to the test sets are withheld. Nevertheless, our strategy achieves a consistent improvement compared with the independent approach and the vector-based conditional approach. The values with a bold type mean the winner across all strategies. As we can observe, the MaP<sub>E</sub> wins 16 out of 16 in both BERT-base and BERT-large groups. It proves that the ensemble of both directions is helpful for the span extraction. In the BiDAF group, The MaP<sub>E</sub> is also the best on all datasets compared with VCP. It shows the robustness of our matrix-based conditional approach in language models. The fact that the MaP<sub>F</sub> wins 12 out of 12 in EM, and 8 out of 12 in F1 demonstrates that the matrix-based conditional approach is capable of predicting a clean answer span that matches human judgment exactly. We suppose the reason is that more start-end position pairs considered in the probability matrix can enhance the interaction and constraint between the start and end, thus, make the MaP<sub>F</sub> perform more consistently in EM than in F1.

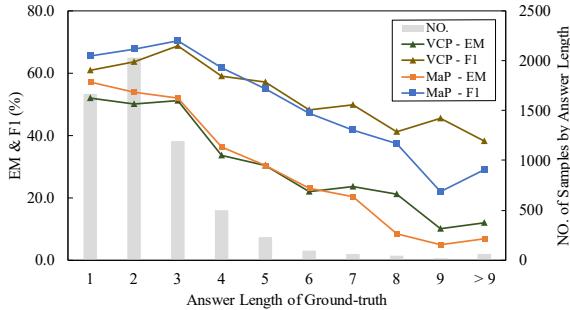


Figure 6: EM and F1 of MaP<sub>F</sub> and VCP based on BiDAF under different answer length.

<sup>3</sup><https://github.com/huggingface/pytorch-transformers>

<sup>4</sup><https://github.com/allenai/allennlp>

### 3.5 Strategy Analysis

Figure 6 shows how the performance changes with respect to the answer length, which is designed on HotpotQA. We can see that the matrix-based conditional approach works better than the vector-based conditional approach as the span decrease in length. Since the short answers have a high rate in all answer spans, so the matrix-based conditional approach is better for the answer span task. In other words, this observation supports the ensemble of both directions as  $E$  does. The MaP<sub>E</sub> combining the MaP<sub>F</sub>'s advantage in short answers and the VCP's advantage in long answers can get a better result than any of them.

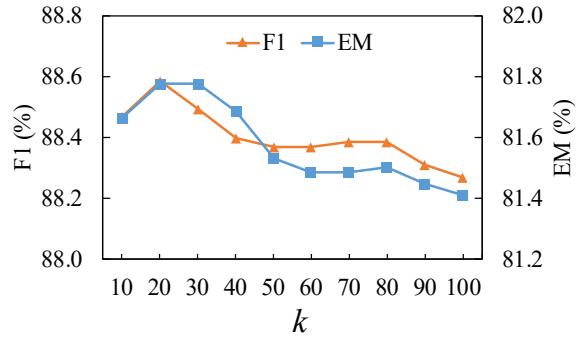


Figure 7: Impact of hyper-parameter  $k$  in MaP<sub>F</sub> on SQuAD 1.1 with BERT-base as the backbone.

We investigate the impact of  $k$  used to choose the top probabilities in the training phase. The results are shown in Figure 7. With the increase of  $k$ , the EM and F1 show a downturn. The best performance happens at  $k = 20$ . We guess that choosing more probabilities makes the training difficult and brings extra noises to the candidate positions. E.g., if  $k$  is set to 30, the number of candidate probabilities will be 900, which is larger than the sequence length 512 in vector-based conditional approach.

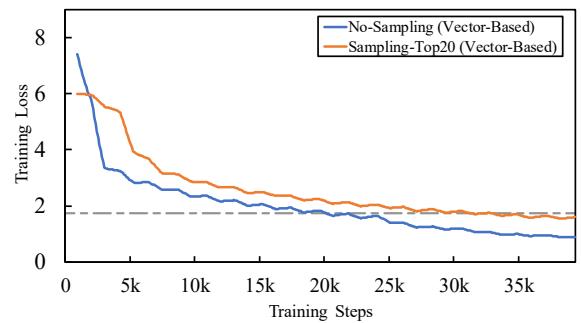


Figure 8: Convergence of sampling-based training strategy on BERT.

We analyze the convergence of the sampling-

based training strategy on SQuAD 1.1. Due to the effectiveness of the sampling-based training strategy is proved in MaP, we conduct an further experiment under the VCP to prove its generalization. Figure 8 demonstrates the results. As our expectation, the sampling-based training strategy optimizes the model as training in whole samples. However, it will cost longer training steps to get the same loss compared with standard training. So our sampling-based training strategy is good for the training of the matrix-based conditional approach.

## 4 Related Work

Machine reading comprehension is an important topic in the NLP community. More and more neural network models are proposed to tackle this problem, including DCN (Xiong et al., 2017), R-NET (Wang et al., 2017), BiDAF (Seo et al., 2017), Match-LSTM (Wang and Jiang, 2017), S-Net (Tan et al., 2018), SDNet (Zhu et al., 2018), QANet (Yu et al., 2018), HAS-QA (Pang et al., 2019). Among various MRC tasks, span extraction is a typical task that extracting a span of text from the corresponding passage as the answer of a given question. It can well overcome the weakness that words or entities are not sufficient to answer questions (Liu et al., 2019).

Previous models proposed for span extraction mostly focus on the design of architecture, especially on the representation of question and passage, and the interaction between them. There are few works devoted to the top-level design of span output, which refers to the probabilities generation from the representation. We divide the previous top-level design into two categories, independent approach and conditional approach. The independent approach is to predict the start and end positions in the given passage independently (Kundu and Ng, 2018a; Yu et al., 2018). Although the independent approach has a simple assumption, it works well when the input features are strong enough, e.g., combining with BERT (Devlin et al., 2019), XL-Net (Yang and Song, 2019), and SpanBERT (Joshi et al., 2019). Nevertheless, since there is a kind of dependency relationship between start and end positions, the conditional approach has advancements over the independent approach.

A typical work on the conditional approach comes from Wang and Jiang (2017). They proposed two different models based on the Pointer Network. One is the sequence model which produces a se-

quence of answer tokens as the final output, and another is the boundary model which produces only the start token and the end token of the answer. The experimental results demonstrate that the boundary model (span extraction) is superior to the sequence model on both EM and F1. The R-NET (Wang et al., 2017), BiDAF (Seo et al., 2017), S-Net (Tan et al., 2018), SDNet (Zhu et al., 2018) have the same output layer and inference phase with the boundary model in (Wang and Jiang, 2017). Lee et al. (2016) presented an architecture that builds fixed length representations of all spans in the passage with a recurrent network to address the answer extraction task. The computation cost is decided by the max-length of the possible span and the sequence length. The experimental results show an improvement on EM compared with the endpoints prediction that independently predicts the two endpoints of the answer span.

However, previous works related to the conditional approach are always based on a probability vector. We investigate another possible matrix-based conditional approach in this paper. Besides, a well-matched training strategy is proposed to our approach, and forward and backward conditional possibilities are also integrated to improve the performance.

## 5 Conclusion

In this paper, we first investigate different approaches of span extraction in MRC. To improve the current vector-based conditional approach, we propose a matrix-based conditional approach. More careful consideration of the dependencies between the start and end positions of the answer span can predict their values better. We also propose a sampling-based training strategy to address the training process of the matrix-based conditional approach. The final experimental results on a wide of datasets demonstrate the effectiveness of our approach and training strategy.

## Acknowledgments

This work was supported by National Key R&D Program of China (2019YFB2101802) and Sichuan Key R&D project (2020YFG0035).

## References

- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of

- deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186.
- Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2019. Spanbert: Improving pre-training by representing and predicting spans. *arXiv:1907.10529*.
- Nitish Shirish Keskar, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Unifying question answering and text classification via span extraction. *arXiv:1904.09286*.
- Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.
- Souvik Kundu and Hwee Tou Ng. 2018a. A nil-aware answer extraction framework for question answering. In *EMNLP*, pages 4243–4252.
- Souvik Kundu and Hwee Tou Ng. 2018b. A question-focused multi-factor attention network for question answering. In *AAAI*, pages 5828–5835.
- Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *TACL*, 7:453–466.
- Kenton Lee, Shimi Salant, Tom Kwiatkowski, Ankur Parikh, Dipanjan Das, and Jonathan Berant. 2016. Learning recurrent span representations for extractive question answering. *arXiv:1611.01436*.
- Shanshan Liu, Xin Zhang, Sheng Zhang, Hui Wang, and Weiming Zhang. 2019. Neural machine reading comprehension: Methods and trends. *arXiv:1907.01118*.
- Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. In *Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016)*, volume 1773.
- Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Lixin Su, and Xueqi Cheng. 2019. HAS-QA: hierarchical answer spans model for open-domain question answering. In *AAAI*, pages 6875–6882.
- Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL-HLT*, pages 2227–2237.
- Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *EMNLP*, pages 2383–2392.
- Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *ICLR*.
- Chuanqi Tan, Furu Wei, Nan Yang, Weifeng Lv, and Ming Zhou. 2018. S-net: From answer extraction to answer generation for machine reading comprehension. In *AAAI*.
- Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. Newsqa: A machine comprehension dataset. In *Rep4NLP@ACL*, pages 191–200.
- Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems*, pages 2692–2700.
- Shuohang Wang and Jing Jiang. 2017. Machine comprehension using match-lstm and answer pointer. In *ICLR 2017: International Conference on Learning Representations, Toulon, France, April 24-26: Proceedings*, pages 1–15.
- Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017. Gated self-matching networks for reading comprehension and question answering. In *ACL*, pages 189–198.
- Caiming Xiong, Victor Zhong, and Richard Socher. 2017. Dynamic coattention networks for question answering. In *ICLR*.
- Liu Yang and Lijing Song. 2019. Contextual aware joint probability model towards question answering system. *arXiv:1904.08109*.
- Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv:1906.08237*.
- Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *EMNLP*, pages 2369–2380.
- Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. 2018. Qanet: Combining local convolution with global self-attention for reading comprehension. In *ICLR*.
- Matthew D. Zeiler. 2012. ADADELTA: an adaptive learning rate method. *arXiv:1212.5701*.
- Chenguang Zhu, Michael Zeng, and Xuedong Huang. 2018. Sdnet: Contextualized attention-based deep network for conversational question answering. *arXiv:1812.03593*.

# Answering Product-related Questions with Heterogeneous Information\*

Wenxuan Zhang, Qian Yu, Wai Lam

The Chinese University of Hong Kong

{wxzhang, yuqian, wlam}@se.cuhk.edu.hk

## Abstract

Providing instant response for product-related questions in E-commerce question answering platforms can greatly improve users' online shopping experience. However, existing product question answering (PQA) methods only consider a single information source such as user reviews and/or require large amounts of labeled data. In this paper, we propose a novel framework to tackle the PQA task via exploiting heterogeneous information including natural language text and attribute-value pairs from two information sources of the concerned product, namely product details and user reviews. A heterogeneous information encoding component is then designed for obtaining unified representations of information with different formats. The sources of the candidate snippets are also incorporated when measuring the question-snippet relevance. Moreover, the framework is trained with a specifically designed weak supervision paradigm making use of available answers in the training phase. Experiments on a real-world dataset show that our proposed framework achieves superior performance over state-of-the-art models.

## 1 Introduction

To help potential consumers address their concerns during online shopping, many E-commerce sites now provide a community question answering (CQA) platform, where users can post questions for a specific product, and others can voluntarily answer them. Very often, it takes a long time for an asker to wait for an answer on such platforms. Therefore, automatically providing a proper response to a product-related question can greatly improve user online shopping experience and stimulate purchase decisions.

\* The work described in this paper is substantially supported by a grant from the Research Grant Council of the Hong Kong Special Administrative Region, China (Project Code: 14200719).

Several efforts have been made to tackle such product-related question answering (PQA) task (McAuley and Yang, 2016; Yu et al., 2018a; Gao et al., 2019; Chen et al., 2019b; Deng et al., 2020b). The existing methods can be generally categorized regarding the involved information source, i.e., from where the responses are obtained. A pioneer work by McAuley and Yang (McAuley and Yang, 2016) investigates answer selection via detecting clues from user reviews. From then on, the review set becomes a commonly used auxiliary information for predicting the answer types or distinguishing true answers from randomly sampled ones (Wan and McAuley, 2016; Yu and Lam, 2018). However, these methods are not feasible for newly-posted questions without candidate answers. A recent approach for PQA task is to directly extract review sentences as the response for a given question (Chen et al., 2019a). But it requires a large number of labeled question-review pairs, whose annotation is a time-consuming and laborious work. Other information sources, such as existing QA collections, are also exploited (Yu et al., 2018b), but relevant QA pairs are assumed to be always available for a new question in their setting, which is uncommon in practice.

Besides user reviews, another kind of information, namely product details provided by the manufacturer are always available and can be an important information source for addressing product-related questions. For example, considering the question "*How large is the keyboard*" for the product shown in Figure 1, the attribute-value pair "*Item Dimensions: 10.9×4.8×0.6 in*" from the specification table can be a good response. Such information can be essential for questions looking for factual type information due to their reliability and precision, but they are often underutilized in previous works. The above scenario motivates our task of answering product-related questions via exploit-

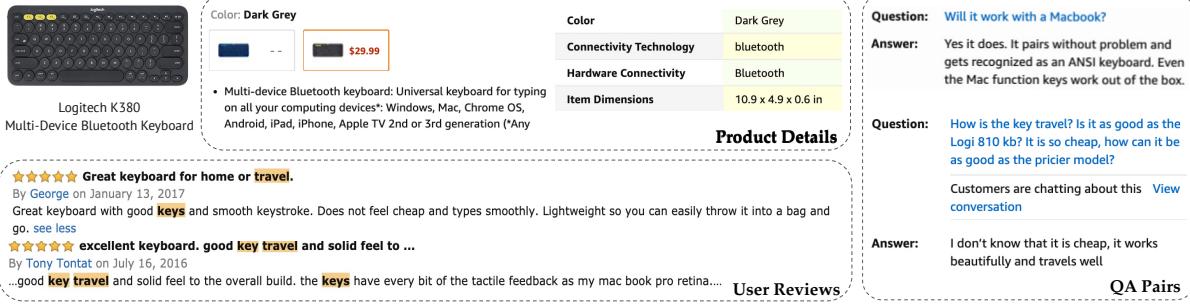


Figure 1: A sample E-commerce product associated with its product details, user reviews, and QA pairs

ing the information from both **product details** and **user reviews** to obtain relevant snippets serving as responses for improving user satisfaction.

This task presents some new research challenges: (i) The heterogeneity of candidate information needs to be appropriately handled. From the above example, we can see that there exists both attribute-value pairs and natural language texts as candidate responses, which implies that typical answer selection approaches (Tan et al., 2016; Wang et al., 2017; Rao et al., 2019) are incapable of handling the concerned task. (ii) Product details and user reviews contain different types of information, which are suitable for answering questions with different information needs. Returning to the example in Figure 1, considering a more subjective question asking about user experience “*How is the key travel*”, snippets from reviews such as “...good key travel and solid feel..” can provide more appropriate responses. Thus, we can observe that questions with different intents can be better answered by snippets from different sources, which should be exploited when measuring the question-snippet relevance. (iii) Training a model to capture the relevance between a question and a candidate snippet with typical supervised paradigms requires a large volume of labeled data. However, it is very time-consuming to manually label the question-snippet pairs in the PQA task due to the product-specific nature of questions and candidate snippets (Chen et al., 2019a), which demands a better solution for training such models.

To tackle these challenges, we propose a novel framework for the PQA task using Heterogenous Information via a Weak Supervision paradigm (HIWS). Given a product-related question, HIWS exploits the corresponding product details and user reviews to return a ranked snippet list serving as the response. Specifically, a heterogeneous infor-

mation encoding component is first developed to encode different information formats into a unified representation composed of a free text sentence and a set of focused aspects. Then for measuring the question-snippet relevance, a gated fusion approach is designed to get aspect-enhanced representations. Also, a question intent analysis module is designed to better determine which information source is more suitable for providing responses. To handle the shortage of labeled data for model training, we develop a weak supervision paradigm making use of the original user-posted answers during training. Some external resources including pre-trained language models such as BERT (Devlin et al., 2019) are utilized to obtain weak supervision signals to facilitate the training process.

Our main contributions are as follows:

- We explore to utilize heterogeneous information including attribute-value pairs and natural language sentences from both product details and user reviews to tackle the PQA task.
- To handle the lack of labeled data, we design an effective weak supervision paradigm making use of available answers in training phase.
- Experiments on real-world E-commerce dataset show that our proposed model achieves superior performance over state-of-the-art models.

## 2 The Proposed Framework

For a product  $p$ , its associated information can be represented as a tuple  $\mathcal{C}_p = (\mathcal{A}, \mathcal{D}, \mathcal{R})$ , where  $\mathcal{A} = \{(a_i, v_i)\}$  is a set of attribute-value pairs extracted from the corresponding specification table.  $\mathcal{D} = \{d_i\}$  denotes the textual product description snippets represented by  $d_i$ ,  $\mathcal{R} = \{r_i\}$  denotes the review set composed of review snippets represented by  $r_i$ . Now given a question  $q$  regarding the product  $p$ , our task is to automatically rank the candidate

snippets in  $\mathcal{C}_p$ , which can either be a textual sentence from  $\mathcal{D}$  or  $\mathcal{R}$ , or an attribute-value pair from  $\mathcal{A}$  for providing responses to the question  $q$ .

As shown in Figure 2, HIWS mainly consists of three components: heterogeneous information encoding, question-snippet relevance matching, and automatic label construction. Concretely, the candidate snippets are first transformed into unified representations. Then we measure the question-snippet relevance both from their aspect-enhanced representations and the intent matching. The overall model is then trained using the automatically-constructed labels via making use of the original answer to the given question.

## 2.1 Heterogeneous Information Encoding

**Heterogeneous Information Unification** Given the heterogeneous candidate snippets including natural language sentences and attribute-value pairs, we transform them into unified representations. It can be observed that these two types of information are actually complementary to each other where the attribute term in an attribute-value pair can well indicate the major focus of such snippet, while a textual sentence can usually provide more detailed semantic information.

To highlight the focus of a natural language sentence  $\bar{c} \in \mathcal{D} \cup \mathcal{R}$ , we can extract  $m$  aspect terms:

$$c^a = \{c_1^a, c_2^a, \dots, c_m^a\} = \text{AE}(\bar{c}) \quad (1)$$

where  $\text{AE}(\cdot)$  refers to a reasonable aspect extraction algorithm such as (He et al., 2017) used in our experiments.  $c^a$  are the extracted  $m$  aspects. These extracted aspects are typically not exactly the same as the terms in the attribute set, but they play a similar role as characterizing the focus of the candidate snippet.

For an attribute-value pair  $(a_i, v_i) \in \mathcal{A}$ , since the main focus of such a snippet is already highlighted by the attribute term  $a_i$ , we directly treat  $a_i$  as the aspect  $c^a$  and construct a pesudo-sentence  $c^t$  by concatenating the attribute and value terms. To this end, any raw snippet  $\hat{c} \in \mathcal{C}_p$ , regardless of its original information type (i.e., whether it is an attribute-value pair or a natural language sentence), is mapped to a unified representation, denoted as  $c$ , as follows:

$$c = (c^t, c^a), \text{ where } c^a = \{c_1^a, c_2^a, \dots, c_m^a\} \quad (2)$$

where  $c^t$  is the textual sentence of  $\hat{c}$ . Such a unified representation facilitates effective processing of

different input formats and also enriches the input representation for later process.

**Snippet Encoding** We next encode the unified candidate snippet representation  $c$  and the question  $q$  to vector representations. We first employ an embedding layer to transform each word into their corresponding word vector. The embedding of the word  $w$  is denoted as  $e_w = [e_w^c; e_w^g]$ , which is a concatenation of character-level embedding  $e_w^c$  and word-level embedding  $e_w^g$ . A bidirectional long short-term memory (Bi-LSTM) network is then employed to encode the local context information for each word in the question and the textual sentence  $c^t$  of the candidate snippet, which generates the context-aware question and snippet representations as follows:

$$h_i^q = \text{Bi-LSTM}(e_i^q, h_{i-1}^q), i \in [1, l_q] \quad (3)$$

$$h_i^c = \text{Bi-LSTM}(e_i^c, h_{i-1}^c), i \in [1, l_c] \quad (4)$$

where  $h_i^*$  is the hidden state of the encoder at the  $i$ -th time step.  $l_q$  and  $l_c$  are the length of the corresponding sequence. We denote the context-aware question and snippet representation as  $H^q \in \mathbb{R}^{l_q \times d_h}$  and  $H^c \in \mathbb{R}^{l_c \times d_h}$  respectively, where  $d_h$  is the number of hidden units of the LSTM network.

Besides the free text part, there are also  $m$  aspects for each candidate snippet  $c$ . They are useful when measuring the relevance between  $q$  and  $c$  since they can be regarded as the most salient part of the candidate snippet. Unlike a textual sentence, aspect terms are often quite short, so we directly employ the character-level embedding to transform each aspect term  $c_i^a$  to a vector representation denoted as  $h_i^a$ :

$$h_i^a = e_{c_i^a}^c = \text{MaxPool}(\text{Conv}(c_i^a)) \quad (5)$$

where  $\text{MaxPool}(\cdot)$  and  $\text{Conv}(\cdot)$  denote the max-pooling and convolutional operations (Kim, 2014).

## 2.2 Question-Snippet Relevance Matching

**Aspect-enhanced Representations** To utilize the aspect information, we design a gated attention mechanism to highlight the relevant information in the question  $q$ . Specifically, for the  $k$ -th word in the context-aware question representation, denoted as  $H_k^q$ , we measure the relative importance  $\alpha_{k[i]}$  of this word given the  $i$ -th aspect term:

$$\alpha_{k[i]} = \frac{\exp((H_k^q)^T h_i^a)}{\sum_{j=1}^{l_q} \exp((H_j^q)^T h_i^a)} \quad (6)$$

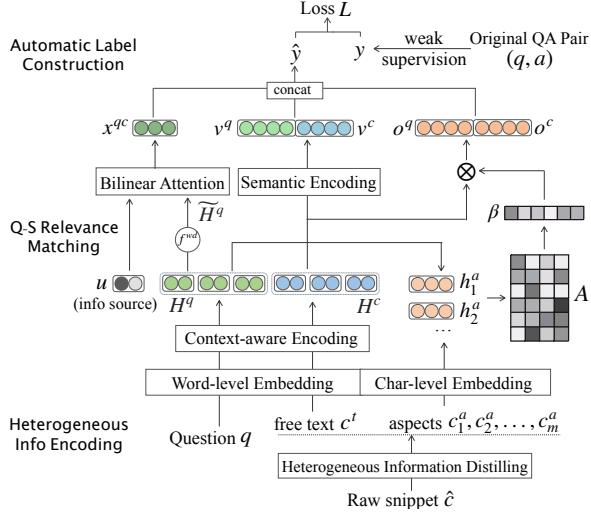


Figure 2: The architecture of proposed HIWS model

Since there are in total  $m$  aspects for a given candidate snippet  $c$ , we can similarly obtain  $\alpha_{k[1]}, \alpha_{k[2]}, \dots, \alpha_{k[m]}$  attention scores for the  $k$ -th question word. These attention scores reflect different relative associations of the concerned word with different aspects. Then for every word in the question  $q$ , we can obtain these attention scores, giving us an attention matrix  $A \in \mathbb{R}^{l_q \times m}$ . To get one composite attention weight for each word in the question, we apply a gated fusion approach to combine these aspects. Specifically, a linear transformation is employed as a gate to learn an appropriate combination between these different attention weights as follows:

$$\beta = \tanh(W_a A^T + b_a) \quad (7)$$

where  $\beta \in \mathbb{R}^{l_q}$  denotes the relative importance of each word in the question  $q$ ,  $W_a$  and  $b_a$  are trainable parameters. Then we can utilize the combined attention weight to obtain an aspect-enhanced question representation  $o^q$ :

$$o^q = \sum_{k=1}^{l_q} H_k^q \cdot \beta_k \quad (8)$$

Here  $o^q$  represents the question representation with an enhancement from multiple aspects of the candidate snippet, which captures the relevance information between  $q$  and  $c$  from the view of aspect terms. Based on the intuition that explicitly highlighting these aspects in  $c^t$  is also helpful to capture its major information, we apply similar operations to  $H^c$ , giving an aspect-aware snippet representation  $o^c$ .

### Question Intent Analysis for Multi-source Candidate Information

The question intent helps

identify what type of information the user is looking for and how to respond them. For example, it can be much more helpful to respond a question asking about personal experience with snippets from reviews. In contrast, the product details will be more suitable and convincing for a question looking for concrete product specifications. Thus, a question intent matching module is designed to detect such matching signals.

It can be observed that the beginning words of a question often have stronger ability for indicating the question intent. Thus, given the question representation  $H^q$ , a weight decay function  $f^{wd}()$  is applied on it to emphasize the importance of the beginning words. Precisely, for the  $i$ -th word in the question, we multiply  $H_i^q$  by  $n^i$ , where  $n \in (0, 1)$  can be set in advance such as  $n = 0.9$  used in our experiments or learned with the model. Then we can obtain the encoded question representation  $r^q$  as follows:

$$\tilde{H}_i^q = f_i^{wd}(H_i^q) = n^i \otimes H_i^q \quad (9)$$

$$r^q = \sum_{i=1}^{l_q} \tilde{H}_i^q \quad (10)$$

where  $\otimes$  refers to the element-wise multiplication. We denote the question representation after such transformation as  $r^q$ . Then given a one-hot feature vector  $u \in \mathbb{R}^2$  of the candidate snippet  $c$  indicating its information source i.e., from product details or user reviews. A bilinear attention layer is employed to achieve the question intent matching analysis:

$$x^{qc} = \tanh(r^q W_m u + b_m) \quad (11)$$

where  $W_m$  and  $b_m$  are trainable parameters,  $x^{qc}$  denotes a low-dimensional vector reflecting the intent matching between the question and the candidate snippet.

### Matching Signal Aggregation and Prediction

After obtaining the aspect-enhanced representations and the question intent matching signals, we also employ a Siamese architecture to encode  $H^q$  and  $H^c$  with another Bi-LSTM encoder for capturing their main semantic information:

$$v^q = \text{Bi-LSTM}_{l_q}(H^q) \quad (12)$$

$$v^c = \text{Bi-LSTM}_{l_c}(H^c) \quad (13)$$

We use  $l_*$  as the subscripts in the above equations to differentiate it from Equation (3) indicating that only the last hidden state is taken as the encoded

representation. By utilizing the same sentence encoder, it helps map them into the same semantic space for determining their semantic relevance.

Then these different matching signals can be aggregated and fed to a MLP layer to make the final prediction  $\hat{y}$ :

$$\hat{y} = \text{MLP}([v^q; v^c; o^q; o^c; x^{qc}]) \quad (14)$$

where the aggregated vector contains matching features from different perspectives including the core semantic information  $v^q$  and  $v^c$ , the aspect-enhanced representations  $o^q$  and  $o^c$  which highlight the major focuses discussed in each sequence, as well as the question intent matching signals  $x^{qc}$  containing information about which information source is better for answering the concerned question regarding its intent.

The overall model is then trained to minimize the cross entropy loss between the predicted relevance score  $\hat{y}$  and the automatically-constructed label  $y$  which will be introduced in the next section:

$$L = -\frac{1}{N} \sum_{n=1}^N [\hat{y}_n \log y_n + (1 - \hat{y}_n) \log (1 - y_n)] \quad (15)$$

where  $\hat{y}_n$  and  $y_n$  denote the prediction and label of the  $n$ -th training instance,  $N$  is the total number of training instances.

### 2.3 Automatic Label Construction

In order to learn a matching function between the question and candidates, the most typical approach is to utilize a large number of annotated sentence pairs (Chen et al., 2019a) to conduct the training. However, this manual solution is not effective in PQA settings due to the large volume of candidate snippets and the product-specific nature of questions and candidates. Fortunately, we can take advantage of the original user-posted answers to their corresponding questions via a weak supervision paradigm during the training phase which has been successfully applied to provide imperfect labels but with far more less human efforts in many NLP tasks such as knowledge-base completion (Hoffmann et al., 2011) and sentiment analysis (Severyn and Moschitti, 2015b) etc.

Given a question  $q$ , we have its answer  $a$  during the training phase as auxiliary information to obtain the label  $y$  for the candidate snippet  $c$ . To make use of the information of the whole QA pair, the entire QA pair  $(q, a)$  is first fused to an integrated textual

snippet  $p^{qa}$  with some heuristic rules (details are given in Sec 3.3). Then the problem of obtaining the relevance label between  $c$  and  $q$  are cast as measuring the relation between  $c^t$  with  $p^{qa}$ . We measure such relation from two perspectives, namely, syntactic relevance and semantic relevance.

**Syntactic Relevance.** Word overlapping between two text items can be a strong signal indicating their relevance. Here we adopt the idea of ROUGE (Lin, 2004) which is initially proposed for computing a recall-based word overlapping score to compute the syntactic-level relevance score  $s_1$ :

$$s_1 = \text{ROUGE-1}(c^t, p^{qa}) + \text{ROUGE-2}(c^t, p^{qa}) \quad (16)$$

where ROUGE-N refers to the overlap of N-grams between  $c^t$  and  $p^{qa}$ .

**Semantic Relevance.** To address the issue of the semantic gap between two text items, many word and sentence embedding models have been proposed and successfully applied to many NLP tasks recently. Here, we utilize some pre-trained text embedding models to compute the semantic relevance between the integrated QA snippet  $p^{qa}$  and  $c^t$ :

$$s_i = \cos(\text{Pre-TE}_i(p^{qa}), \text{Pre-TE}_i(c^t)) \quad (17)$$

where Pre-TE refers to a pre-trained text encoder. We adopt GloVe (Pennington et al., 2014), Elmo (Peters et al., 2018) and BERT (Devlin et al., 2019) in our experiments.  $\cos(\cdot)$  denotes the cosine similarity score between the two encoded sentence representations. We denote the computed relevance scores with the aforementioned pre-trained models as  $s_2, s_3, s_4$  respectively.

After obtaining these relevance signals, a small amount of human-annotated question-snippet pairs are used to train a simple classifier for learning to combine these signals into the single label  $y^1$ . Note that it seems to be unnecessary to design any framework if a simple classifier with a few amount of labeled data and some pre-trained models can achieve a high accuracy. This is because we use the information of the entire QA pair to obtain the label  $y$  denoting the question-snippet relevance, which is different when we only have the question  $q$  and needs to retrieve relevant snippets during the testing phase. Thus a simple classifier with a few amount of labeled data can learn to integrate these relevance scores for the construction of “gold” labels with the help of original answers.

---

<sup>1</sup>40 questions with their candidate snippets are annotated for this purpose, a SVM classifier is used in our experiment.

### 3 Experiments

#### 3.1 Dataset

We perform experiments on real-world data to validate the model effectiveness. The question-answer pairs and reviews are drawn from the Amazon QA dataset (McAuley and Yang, 2016) and Amazon review dataset (Ni et al., 2019). Product details are crawled from the corresponding products’ pages and incorporated into our dataset. In this way, we construct a heterogeneous dataset, which includes in total 5,395 QA pairs of 3,840 products spanning three product categories, namely, “*Cell Phones and Accessories*”, “*Sports and Outdoors*” and “*Tools and Home Improvement*”.

For each question, we first utilize the BM25 algorithm to conduct an initial filtering and collect the 50 top-ranked snippets from the corresponding product information as candidate snippets. After discarding empty or meaningless strings, we obtain 219,563 question-candidate snippet pairs in total. The dataset is split for training/validation/testing as 4,023 / 779 / 593 questions respectively, which results in 163,063 / 32,178 / 24,322 question-snippet pairs in each set. To obtain training and validation set, we utilize the weak supervision paradigm described in Sec 2.3 to automatically construct labels. For the testing set, in order to evaluate the effectiveness of the whole framework, the relevance labels between the questions and candidate snippets are annotated manually by two trained human annotators, the disagreements of the annotations are resolved by another experienced annotator

#### 3.2 Baselines and Evaluation Metrics

To compare with our proposed framework, we adopt several strong baseline and state-of-the-art question answering models, including **CNN** (Severyn and Moschitti, 2015a), **QA-LSTM** (Tan et al., 2016), **MatchPyramid** (Pang et al., 2016), **BiMPM** (Wang et al., 2017), **Conv-KNRM** (Dai et al., 2018), **HCAN** (Rao et al., 2019) for comparisons. These models take the question and natural language sentence part of the candidate snippet as input, and are trained using the same automatically-constructed labels derived from original QA pairs as our proposed HIWS framework.

Two retrieval-based unsupervised models are also adopted: (1) **BM25**: It is a widely-used bag-of-words retrieval model. (2) **QCEM**: Question Candidate Embedding Matching is an unsupervised method that sums the word vectors of each sentence

Table 1: Response Selection Performance

	MAP	MRR	P@5	P@10
BM25	0.417	0.549	0.296	0.234
QCEM	0.479	0.623	0.385	0.278
CNN	0.576	0.665	0.430	0.329
QA-LSTM	0.561	0.656	0.419	0.327
MatchPyramid	0.630	0.700	0.466	0.353
BiMPM	0.613	0.683	0.458	0.336
Conv-KNRM	0.615	0.696	0.457	0.337
HGAN	0.632	0.710	0.459	0.339
HIWS	<b>0.674</b>	<b>0.749</b>	<b>0.498</b>	<b>0.363</b>

as the sentence embedding, and cosine similarity is utilized for predicting sentence relevance.

For evaluation metrics, Mean Average Precision (MAP), Mean Reciprocal Rank (MRR), and Precision at N (P@N) are used to measure the performance. Precision at N (P@N) is the precision of the N retrieved snippets. We set N=5 and N=10 which correspond to P@5 and P@10 respectively in our experiments.

#### 3.3 Implementation Details

For the automatic label construction, we first utilize the user-posted answer to paraphrase the question for obtaining the integrated snippet  $p^{qa}$  according to the part-of-speech tags and syntactic structure of the question with heuristic rules. For example, for a question “*does it have a front-facing camera?*” with the answer “*No.*”, it will be combined to “*It does not have a front-facing camera*”.

For the network architecture, we initialize the word embedding layer with the pre-trained 300D GloVe word vectors (Pennington et al., 2014). The sizes of the CNN filters in the character-level embedding are set to [2, 3, 4, 5], each with 75 filters, resulting in 300D character-level embedding for each word. The hidden dimension of the context-aware Bi-LSTM encoder is set to 150, with the dropout rate being 0.3. The hidden dimension of the sentence encoder in Eq. (12) is set to 64, with the dropout rate also being 0.3. The hidden dimensions of the MLP layer in the final prediction layer are set to 300 and 100 respectively, with ReLU as the activation function. All models are trained with the batch size of 100. The number of aspects  $m$  for each candidate snippet is set to be 3 which is a moderate number for a single sentence.

Table 2: Effectiveness of Weak Supervision Paradigm

	BiMPM		HCAN		HIWS	
	MAP	MRR	MAP	MRR	MAP	MRR
with QA	0.338	0.409	0.329	0.402	0.310	0.393
with SQS	0.443	0.492	0.432	0.495	0.479	0.556
with WS	<b>0.613</b>	<b>0.683</b>	<b>0.632</b>	<b>0.710</b>	<b>0.674</b>	<b>0.749</b>

### 3.4 Quantitative Evaluation Results

**Response Selection Performance** The evaluation results are presented in Table 1, which demonstrates that our proposed HIWS achieves the best performance among all evaluation metrics compared with both retrieval-based solutions and supervised QA matching methods. We can observe that some simple QA models such as QA-LSTM and unsupervised models such as QCEM can still achieve reasonable performance. For those state-of-the-art models such as BiMPM and HCAN, although equipped with complicated network architecture, they do not perform as promising as expected. Such a result is due to the fact that these QA models merely focus on the matching between text items and ignore some important characteristics in the E-commerce scenario such as the heterogeneous information formats and multiple information sources of the candidate snippets. HIWS exploits such characteristics and utilize the extracted aspects to obtain enriched representations, leading to its superior performance.

**Effectiveness of Proposed Weak Supervision Paradigm** We investigate two alternative strategies for tackling the shortage of labeled data and compare them with our proposed weak supervision strategy to examine its effectiveness. The results on the same test set are reported in Table 2. Specifically, we train HIWS and two baselines, namely BiMPM and HCAN with different methods: “with QA” denotes training with the QA pairs instead of question-snippet pairs as in Table 1. We treat questions with their original answers as the positive samples and other randomly selected answers as negative samples for model training; “with SQS” refers to models which are first trained with QA pairs, then the Small number of annotated Question-Snippet pairs introduced in Sec 2.3 are used to fine-tune the model; “with WS” means the model is trained with the proposed weak supervision approach. Comparing these model variants, we can observe that models trained with the original QA pairs perform quite worse, showing the

Table 3: Ablation study for components in HIWS

Ablation of HIWS	MAP	MRR
w/o syntactic relevance score	0.273	0.434
w/o semantic relevance score	0.543	0.626
w/o question intent matching	0.667	0.737
w/o aspect-enhanced representations	0.631	0.704
HIWS	<b>0.674</b>	<b>0.749</b>

semantic gap between the original answers and the candidate snippets needs to be handled properly. Models with SQS outperform models with QA via fine-tuning with proper data, but it still failed to achieve satisfactory results due to the limited amount of labeled data. However, performance for all models can be improved with our proposed weak supervision paradigm, demonstrating its effectiveness on utilizing original answer information for bridging the connection between the question and snippets in the E-commerce settings.

**Ablation Analysis** We conduct ablation analysis to investigate the effectiveness of some important components in HIWS as shown in Table 3. We first create two sets of training labels whose construction step only involves one kind of relevance scores introduced in Sec 2.3, denoted as “w/o syntactic relevance score” and “w/o semantic relevance score” respectively. It can be observed that these two kinds of linguistic considerations, especially the syntactic relevance, are quite essential for automatically obtaining the labels for conducting training and thus directly influence the final performance of our model. Another two important components in HIWS are the aspect-enhanced representations and the question intent matching. As shown in Table 3, these two components contribute to some performance boost, especially the aspect-enhanced module. For constructing the variant model without aspect-enhanced representations, we still feed the embedded aspect  $h_i^a$  into the aggregation layer. Thus, even without considering the interaction between aspects and the question as in HIWS, this variant still outperforms some baselines.

**Performance with Different Amount of Data** We further investigate the robustness of HIWS via examining its performance with different amount of training data. The MAP and MRR scores under each product category are reported in Figure 3, where “w/ n data” refers to HIWS trained with  $n$  proportion of the entire training data. It can be observed that even when we use a moderate amount of training data such as 3/4 training data, the per-

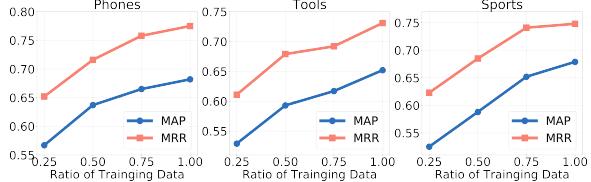


Figure 3: Performance with Different Amount of Data

formance does not drop significantly. Such results show the robustness of our proposed model implying that it can effectively utilize the available QA pairs to automatically construct useful training signals for learning the question-snippet relevance relation.

### 3.5 Case Study

To gain some insights into HIWS, we present two sample questions with the top-one responses given by HIWS and two strong existing methods in Table 4. The information sources of each snippet are marked, where  $\mathcal{A}, \mathcal{D}, \mathcal{R}$  refers to attribute-value pairs, product textual descriptions and reviews respectively. Following each information source symbol, the correctness of the retrieved response is given. From the results, we can observe that HIWS successfully handles candidate snippets from different sources to answer the product questions with different information needs. For example, it precisely retrieves the corresponding attribute of the product for Question-1, which is more reliable and precise than the snippet retrieved from the review set by the existing models. Moreover, HIWS correctly handles the second question while the focused aspect is missing in responses from other methods. This is likely due to the aspect-enhanced representations for highlighting the major focus in the question and snippets. This result shows the necessity of effectively exploring different types of information of the concerned product instead of considering a single information source as in previous works.

## 4 Related Work

In recent years, many deep learning based methods have been proposed for the answer selection task in community question answering (CQA) platforms. These models can be generally categorized into two types according to their network architecture (Lai et al., 2018), namely Siamese networks (Tan et al., 2016; Mueller and Thyagarajan, 2016) and Compare-Aggregate networks (Wang

Table 4: Two sample questions with the top-one responses returned by HIWS and two existing models. The information sources and the gold labels of the snippets are also marked out in the parentheses at the end of each snippet respectively.

**Question-1:** What is the overall length of this bulb ?

**HIWS:** Product Dimensions : 6.5 x 2.5 x 2.5 inches ( $\mathcal{A}$ ) (✓)

**MatchPyramid:** I decided to try using these before i went more expensive route, the bulb are indeed quite large the length of a hand perhaps. ( $\mathcal{R}$ ) (✗)

**HCAN:** I will update this review to render my durability opinion, one last note pay attention to the length of these bulb. ( $\mathcal{R}$ ) (✗)

**Question-2:** Will this work with my unlocked fire phone i have straight talk i want to switch to the amazon fire phone.

**HIWS:** Sim card will only work with an att compatible or unlocked gsm phone ( $\mathcal{D}$ ) (✓)

**MatchPyramid:** Keep your current phone number. Works with SIMs, IM, social networks, email, and web. ( $\mathcal{D}$ ) (✗)

**HCAN:** I have tmobile and the service is not good in my area so i want to switch to straight talk ( $\mathcal{R}$ ) (✗)

et al., 2017; Rao et al., 2019; Deng et al., 2020a).

Product-related Question Answering (PQA) problem has drawn a lot of attention recently, due to the increasing popularity of online shopping. Most of the existing works utilize reviews as their major information to provide responses for a given question. McAuley and Yang (2016) treat reviews as “experts” to handle the answer selection task. Later, product aspects are considered to further improve the performance (Yu and Lam, 2018). Chen et al. (2019a) propose to tackle PQA task by directly retrieving review sentences as answers. However, it requires a large number of labeled question-review pairs. Yu et al. (2018b) assume that relevant QA pairs are always available for a given question which can be utilized to provide the responses. Some other works formulate the PQA task as a reading comprehension problem (Xu et al., 2019), where the main focus is to extract a text span as the answer given a relevant review, which is unavailable in many cases. Given some successful applications of text generation models such as text summarization (Rush et al., 2015) and response generation (Tao et al., 2018), some models are proposed to generate an answer sentence (Gao et al., 2019; Chen et al., 2019b) given relevant product information, some later works specifically consider the user opinion information during such generation process (Deng et al., 2020b). Since most product-related questions are looking for diverse answers, we argue that information extracted from reliable sources is more effective and explainable

solution for the PQA task. More recently, some studies consider the answer helpfulness prediction task (Zhang et al., 2020b) and answer ranking problem (Zhang et al., 2020a) in the context of PQA, assuming the existence of user-provided answers to a given question. Different from them, we aim to provide instant responses for a newly-posted question in E-commerce.

## 5 Conclusions

We propose a novel framework for answering product-related questions via exploiting heterogeneous information including attribute-value pairs and free text sentences from both product details and user reviews. To tackle the shortage of labeled data, we design a weak supervision paradigm by making use of the existing QA pairs to automatically construct labels for training. Extensive experiments conducted on a real-word dataset demonstrate the superiority of our proposed framework.

## References

- Long Chen, Ziyu Guan, Wei Zhao, Wanqing Zhao, Xiaopeng Wang, Zhou Zhao, and Huan Sun. 2019a. Answer identification from product reviews for user questions by multi-task attentive networks. In *AAAI*, pages 45–52.
- Shiqian Chen, Chenliang Li, Feng Ji, Wei Zhou, and Haiqing Chen. 2019b. Review-driven answer generation for product-related questions in e-commerce. In *WSDM*, pages 411–419.
- Zhuyun Dai, Chenyan Xiong, Jamie Callan, and Zhiyuan Liu. 2018. Convolutional neural networks for soft-matching n-grams in ad-hoc search. In *WSDM*, pages 126–134.
- Yang Deng, Wai Lam, Yuexiang Xie, Daoyuan Chen, Yaliang Li, Min Yang, and Ying Shen. 2020a. Joint learning of answer selection and answer summary generation in community question answering. In *AAAI*, pages 7651–7658.
- Yang Deng, Wenxuan Zhang, and Wai Lam. 2020b. Opinion-aware answer generation for review-driven question answering in e-commerce. In *CIKM*.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, pages 4171–4186.
- Shen Gao, Zhaochun Ren, Yihong Zhao, Dongyan Zhao, Dawei Yin, and Rui Yan. 2019. Product-aware answer generation in e-commerce question answering. In *WSDM*, pages 429–437.
- Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2017. An unsupervised neural attention model for aspect extraction. In *ACL*, pages 388–397.
- Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *ACL*, pages 541–550.
- Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *EMNLP*, pages 1746–1751.
- Tuan Manh Lai, Trung Bui, and Sheng Li. 2018. A review on deep learning techniques applied to answer selection. In *COLING*, pages 2132–2144.
- Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.
- Julian McAuley and Alex Yang. 2016. Addressing complex and subjective product-related queries with customer reviews. In *WWW*, pages 625–635.
- Jonas Mueller and Aditya Thyagarajan. 2016. Siamese recurrent architectures for learning sentence similarity. In *AAAI*, pages 2786–2792.
- Jianmo Ni, Jiacheng Li, and Julian J. McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *EMNLP-IJCNLP*, pages 188–197.
- Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2016. Text matching as image recognition. In *AAAI*, pages 2793–2799.
- Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543.
- Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL-HLT*, pages 2227–2237.
- Jinfeng Rao, Linqing Liu, Yi Tay, Wei Yang, Peng Shi, and Jimmy Lin. 2019. Bridging the gap between relevance matching and semantic matching for short text similarity modeling. In *EMNLP-IJCNLP*, pages 5373–5384.
- Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *EMNLP*, pages 379–389.
- Aliaksei Severyn and Alessandro Moschitti. 2015a. Learning to rank short text pairs with convolutional deep neural networks. In *SIGIR*, pages 373–382.
- Aliaksei Severyn and Alessandro Moschitti. 2015b. Twitter sentiment analysis with deep convolutional neural networks. In *SIGIR*, pages 959–962.

Ming Tan, Cícero Nogueira dos Santos, Bing Xiang, and Bowen Zhou. 2016. Improved representation learning for question answer matching. In *ACL*.

Chongyang Tao, Shen Gao, Mingyue Shang, Wei Wu, Dongyan Zhao, and Rui Yan. 2018. Get the point of my utterance! learning towards effective responses with multi-head attention mechanism. In *IJCAI*, pages 4418–4424.

Mengting Wan and Julian McAuley. 2016. Modeling ambiguity, subjectivity, and diverging viewpoints in opinion question answering systems. In *ICDM*, pages 489–498.

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017. Bilateral multi-perspective matching for natural language sentences. In *IJCAI*, pages 4144–4150.

Hu Xu, Bing Liu, Lei Shu, and Philip S. Yu. 2019. BERT post-training for review reading comprehension and aspect-based sentiment analysis. In *NAACL-HLT*, pages 2324–2335.

Jianfei Yu, Minghui Qiu, Jing Jiang, Jun Huang, Shuangyong Song, Wei Chu, and Haiqing Chen. 2018a. Modelling domain relationships for transfer learning on retrieval-based question answering systems in e-commerce. In *WSDM*, pages 682–690. ACM.

Qian Yu and Wai Lam. 2018. Review-aware answer prediction for product-related questions incorporating aspects. In *WSDM*, pages 691–699.

Qian Yu, Wai Lam, and Zihao Wang. 2018b. Responding e-commerce product questions via exploiting qa collections and reviews. In *COLING*, pages 2192–2203.

Wenxuan Zhang, Yang Deng, and Wai Lam. 2020a. Answer ranking for product-related questions via multiple semantic relations modeling. In *ACM SIGIR*, pages 569–578.

Wenxuan Zhang, Wai Lam, Yang Deng, and Jing Ma. 2020b. Review-guided helpful answer identification in e-commerce. In *WWW*, pages 2620–2626.

# Two-Step Classification using Recasted Data for Low Resource Settings

Shagun Uppal<sup>1</sup>, Vivek Gupta<sup>2</sup>, Avinash Swaminathan<sup>1</sup>,  
Debanjan Mahata<sup>3</sup>, Rakesh Gosangi<sup>3</sup>, Haimin Zhang<sup>3</sup>  
Rajiv Ratn Shah<sup>1</sup>, Amanda Stent<sup>3</sup>

<sup>1</sup> IIIT-Delhi, India, <sup>2</sup> University of Utah, <sup>3</sup> Bloomberg, New York  
shagun16088@iiitd.ac.in, vgupta@cs.utah.edu, s.avinash.it.17@nsit.net.in,  
{dmahata, rgosangi, hzhang449, astent}@bloomberg.net,  
rajivratn@iiitd.ac.in

## Abstract

An NLP model’s ability to reason should be independent of language. Previous works utilize Natural Language Inference (NLI) to understand the reasoning ability of models, mostly focusing on high resource languages like English. To address scarcity of data in low-resource languages such as *Hindi*, we use data recasting to create four NLI datasets from existing four text classification datasets in *Hindi* language. Through experiments, we show that our recasted dataset<sup>1</sup> is devoid of statistical irregularities and spurious patterns. We study the consistency in predictions of the textual entailment models and propose a consistency regulariser to remove pairwise-inconsistencies in predictions. Furthermore, we propose a novel *two-step* classification method which uses textual-entailment predictions for classification task. We further improve the classification performance by jointly training the classification and textual entailment tasks together. We therefore highlight the benefits of data recasting and our approach<sup>2</sup> with supporting experimental results.

## 1 Introduction

Textual entailment (TE) is the task of determining if a *hypothesis* sentence can be inferred from a given *context* sentence. Figure 1 shows examples of *context-hypothesis* pairs for TE. Previous works (Wang and Zhang, 2009; Tatú and Moldovan, 2005; Sammons et al., 2010) investigated several semantic approaches for TE and demonstrated how they can be used to evaluate inference-related tasks such as Ques-

tion Answering (*QA*), reading comprehension (*RC*) and paraphrase acquisition (*PA*).

Context-Hypothesis	Label
<i>p</i> : The kid exclaimed with joy.	<i>entailed</i>
<i>h</i> : The kid is happy.	
<i>p</i> : I am feeling happy.	<i>not-entailed</i>
<i>h</i> : I am angry.	(contradictory)

Table 1: Example illustrating context (*c*) - hypothesis (*h*) pairs for the task of textual entailment.

Researchers have curated many resources<sup>3</sup> and benchmark datasets for TE in English (Bowman et al., 2015; Williams et al., 2018; Khot et al., 2018). However, to our knowledge, there is only one TE dataset (XNLI) in Hindi, which was created by translating English data (Conneau et al., 2018) and another in Hindi-English code-switched setting (Khanuja et al., 2020). Hindi is the language with the fourth most native speakers in the world<sup>4</sup>. Despite its wide prevalence, Hindi is still considered a low-resource language by NLP practitioners because there are a rather limited number of publicly available annotated datasets. Developing models that can accurately process text from low-resource languages, such as Hindi, is critical for the proliferation and broader adoption of NLP technologies.

Creating a high-quality labeled corpus for TE in Hindi through crowd-sourcing could be challenging. In this paper, we employ a recasting technique from Poliak et al. (2018a,b) to convert four publicly available text classification datasets in Hindi and pose them as TE problems. In this recasting process, we build template hypotheses for each class in the label taxonomy. Then, we pair the original anno-

<sup>1</sup><https://github.com/midas-research/hindi-nli-data>

<sup>2</sup><https://github.com/midas-research/hindi-nli-code>

<sup>3</sup>[https://aclweb.org/aclwiki/Textual\\_Entailment\\_Resource\\_Pool](https://aclweb.org/aclwiki/Textual_Entailment_Resource_Pool)

<sup>4</sup>[https://en.wikipedia.org/wiki/List\\_of\\_languages\\_by\\_number\\_of\\_native\\_speakers](https://en.wikipedia.org/wiki/List_of_languages_by_number_of_native_speakers)

tated sentence with each of the template hypotheses to create TE samples. Unlike XNLI, our dataset is based on the original Hindi text and is not translated. Furthermore, the multiple annotation artefacts (Tan et al., 2019) present in the original classification data are leveled out for the Textual entailment task on the recasted data due to label balance<sup>5</sup>.

We evaluated state-of-the-art language models (Conneau et al., 2019) performance on the recasted TE data. We then combine the predictions of related pairs (same premise) from TE task to predict the classification labels of the original data (premise sentence), a *two-step classification*. We observed that a better TE performance on the recasted data leads to higher accuracy on the followed classification task. We also observed that TE models can make inconsistent predictions across samples derived from the same *context* sentence. Driven by these observations, we propose two improvements to TE and classification modeling. First, we introduce a regularisation constraint based on the work of (Li et al., 2019) that enforces consistency across pairs of training samples, thus correcting inconsistent predictions. Second, we propose a joint objective for training TE and classification simultaneously. Our results demonstrate that the regularization constraint and joint training helps improve the performance of both the TE models and the followed classification task. Though our work demonstrates the use of recasting and modeling improvements for TE in Hindi, we expect these techniques can be applied to other low-resource languages and other semantic phenomenon beyond textual classification.

Following are the main contributions of this work:

1. We develop new NLI datasets for a low-resource language *Hindi* using recasting (Section 3) and evaluated state-of-the-art language models on them (Section 4.1).
2. Based on our analysis of inconsistencies in the predictions of TE models, we propose a new regularisation constraint (Section 4.1.1).

3. We propose a two-step classification approach that uses TE predictions from *context-hypothesis* pairs to predict the labels of the original classification task (Section 4.2).
4. We propose a novel joint-training objective paired with consistency regularisation to obtain state-of-the-art performance for text classification on four Hindi datasets (Section 4.2.1).

## 2 Related Work

In this section, we list some of the related works in the field of NLI as well as challenges encountered in low-resource settings.

### 2.1 Natural Language Inference

Recent studies in the field of NLI have emphasized the role of TE for estimating language comprehensibility of the models. White et al. (2017) takes into consideration the need to leverage the existing pool of annotated collections as targeted textual inference examples (such as pronoun resolution and sentence paraphrasing). Poliak et al. (2018b) discussed existing biases in NLI datasets which helps the models to perform well on Hypothesis-only baselines. Poliak et al. (2018a) analysed NLI datasets based on various semantic phenomenon to verify the ability of a model to perform unique, varied levels of reasoning. It performs data recasting on existing classification datasets to obtain a conventional context/hypothesis/label for common NLI tasks. Several modifications have been tried over baseline models for enhanced NLI and NLU. Liu et al. (2019) focuses on NLU over cross-task data to achieve generalisability over new unseen tasks. Li et al. (2018) incorporates attention mechanism to capture semantic relations in between individual words of the sentence for robust encodings.

However, NLI has mostly revolved around English language. Our approach is motivated by such studies to analyse NLU using current embeddings for low-resource languages like *Hindi*. Bhattacharyya (2012) discusses some of the key challenges associated with *Hindi*, for example, grammatical constraints for most words to be masculine/feminine (similar to French and unlike English), which makes

<sup>5</sup>See Appendix Section A.4 for other benefits of recasting data.

semantic tasks like pronoun resolution, paraphrasing tough.

## 2.2 NLP for Low-Resource Languages

In a plethora of diverse languages, only a handful of them have plenty of labeled resources for data-driven analysis and advancements (Joshi et al., 2020). Data in low-resource languages is either unlabeled or resides in spoken dialect than texts. There have been recent efforts using curriculum learning for making pretrained language models for several multi-lingual tasks (Conneau et al., 2018, 2019). However, many such languages give rise to creoles, building new mixed languages at the interface of existing languages. One such example is Hinglish (Hindi + English) that has widely been taken over in the form of tweets and social media messages. Attempts have been made to study linguistic tasks like language identification, NER (Singh et al., 2018) and detection of hate speech from social media (Mathur et al., 2018). (Sitaram et al., 2019) looks at the challenges and opportunities of code-switching.

Joshi et al. (2019) compares the current deep learning methods for classification tasks in Hindi and concludes the need of more efficient models for the same. Apart from that, low-resource languages also challenge us to shift from data-driven modelling to intelligent neural modelling. This improves language understanding from limited available data and also diminishes the need of hand-engineered feature representations similar to generative modelling. Some such efforts have been put forth by Kumar et al. (2019) and Akhtar et al. (2016). Keeping these challenges in mind, this work is a step towards understanding of a low-resource language - *Hindi* using TE.

## 3 Recasting Classification Datasets

One of the main challenges for TE evaluation for low-resource languages is the lack of labeled data. In this work, we employ recasting to convert annotated classification datasets in *Hindi* to labeled TE samples. As in (Polliak et al., 2018a), we selected four different datasets for recasting thus introducing linguistic diversity in the resulting TE dataset.

**Product Review** - The first dataset (*PR*) contains 5,417 samples of online user reviews

in Hindi for different products (Akhtar et al., 2016). These samples were annotated into one of the following four sentiment classes: *positive*, *negative*, *neutral*, and *conflict*. For recasting the samples in this dataset, we first built 8 hypothesis templates: 2 per class label. For each label, we create one positive and one negative hypothesis which roughly translate to: ‘*This product got <label> reviews*’ and ‘*This product did not get <label> reviews*’.

Given a sample from the *PR* dataset, we treat it as the context sentence and combine with the 8 hypotheses sentences to create NLI samples. If the *<label>* of the premise matches that of the positive hypothesis, then the NLI sample is marked as ‘*entailed*’. Likewise, if the *<label>* of the premise does not match the negative hypothesis, then the NLI sample is also marked as ‘*entailed*’. For the remaining cases, the sample is marked as ‘*non-entailed*’. This process is summarized with an example in Figure 1. For more detailed recasting illustration, see Appendix Section A.1 Figure 5.

**BHAAV** - The second dataset BHAAV (*BH*) (Kumar et al., 2019) contains 20,304 sentences from Hindi short stories annotated for one of the following five emotion categories: *joy*, *anger*, *suspense*, *sad*, and *neutral*. We used a similar process as *PR* to recast *BH* using the following templates to create the hypothesis: ‘*It is a matter of great <label>*’ and ‘*It is not a matter of great <label>*’.

### Hindi Discourse Modes Dataset (HDA)

- This dataset (Dhanwal et al., 2020) consists of 10,472 sentences from Hindi short stories annotated for five different discourse modes *argumentative*, *narrative*, *descriptive*, *dialogic* and *informative*.

**Hindi BBC News Dataset (BBC)** - This dataset<sup>6</sup> contains 4,335 Hindi news headlines tagged across 14 categories: *India*, *Pakistan*, *news*, *International*, *entertainment*, *sport*, *science*, *China*, *learning english*, *social*, *southasia*, *business*, *institutional*, *multimedia*. We processed this dataset to combine two sets of relevant but low prevalence classes. Namely, we merged the samples from *Pakistan*, *China*, *international*, and *southasia* as one class called

<sup>6</sup><https://tinyurl.com/y8hxtbn8>

● c: Context ● h: Hypothesis ● TE: Textual Entailment

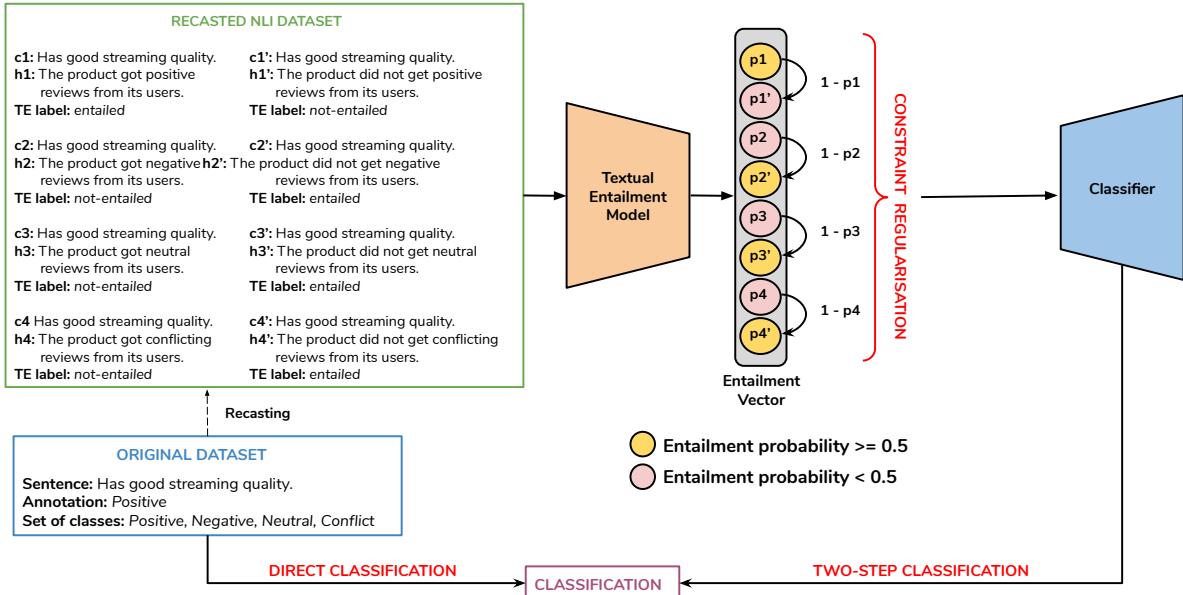


Figure 1: Illustration of the proposed approach

**international.** Likewise, we also merged samples from *news*, *business*, *social*, *learning english*, and *institutional* as **news**. Lastly, we also removed the class *multimedia* because there were very few samples.

Table 2 shows statistics about the datasets and Table 3 shows examples from each.

	Datasets			
	PR	BH	HDA	BBC
<b>Original datasets</b>				
# Classes	4	5	5	6
# Train	4334	16243	8377	3889
# Dev	541	2030	1047	216
# Test	542	2031	1048	217
<b>Recasted TE data</b>				
# Classes	2	2	2	2
# Train	17336	64972	33508	15556
# Dev	4328	20300	10470	2592
# Test	4336	20310	10480	2604

Table 2: Statistics of the original classification data and recasted NLI data.

## 4 Methodology

Our objective in this paper is not only to use recasting to create a NLI dataset in low-resource settings but also to understand how different models are effective in both TE and classification task. Furthermore, we also discuss our novel two-step classification technique with joint objective and regularization constraints.

### 4.1 Textual Entailment

One straightforward application of NLI comes with evaluating the task of Textual Entailment

(TE). It analyses if the TE model can draw reasonable inferences from the context to hypothesise over other related/unrelated data, as shown in Table 1.

However, apart from being correct/incorrect, certain times, TE models are not always consistent with their own beliefs (Li et al., 2019) due to spurious patterns in the dataset (Poliak et al., 2018a). Consider two *context-hypothesis* pairs  $P$  and  $P'$  generated from the same context sentence and opposing hypotheses statements (as illustrated in Figure 1). Consequently,  $P$  and  $P'$  would have opposing TE labels. When a TE model makes predictions on these two pairs, there are three possibilities (Table 5). The model can get both predictions right, in which case the predictions are consistent. It can also get both predictions wrong but still they are consistent. Lastly, it can get one of the predictions wrong, in which case they are inconsistent<sup>7</sup>. To mitigate this inconsistency problem, we propose consistency regularization loss.

#### 4.1.1 Consistency Regularisation (CR)

To enforce this pairwise-consistency, we add a regularization loss<sup>8</sup>, inspired from (Li et al.,

<sup>7</sup>See Appendix Section A.3 Table 11 for additional inconsistency examples.

<sup>8</sup>Other suitable loss function also works (Li et al., 2019).

Dataset	Sentence (Hindi)	Sentence (English)	Sentiment
PR	फिल्हाल , इसमें कोई वीडियो या वॉयस कॉल सपोर्ट नहीं है।	At the moment, there is no video or voice call support.	<i>negative</i>
BH	इतनी मिठाइयाँ ली, मुझे किसी ने एक भी न दी।	Took so many sweets, nobody gave me one.	<i>anger</i>
HDA	सौर मंडल के सारे ग्रह ब्रह्मस्पति में समां सकते हैं।	All the planets in the solar system can be contained within the Jupiter.	<i>informative</i>
BBC	अखबार ने बताया कि फेसबुक पर मिलेगी असल जादू की झाप्पी।	The newspaper said that real magic hug will be found on Facebook.	<i>entertainment</i>

Table 3: Sample sentences from the four datasets and the corresponding annotation labels.

2019), for our settings, where the entailment probabilities  $p$  and  $p'$  of pairs  $P$  and  $P'$  respectively, is required to always sum up to one as illustrated in Figure 1. Mathematically, we define the regularisation term as depicted in Equation 1.

$$\mathcal{L}_{reg} = \|p + p' - 1\|_2^2 \quad (1)$$

Our regularisation is different from (Li et al., 2019) in terms of different consistency problem being considered, which in-term diversifies a very different inductive bias from former.

#### 4.2 Two-step classification

We further extend the knowledge accumulated by TE predictions for multi-class classification. Consider a TE model with binary output where 1 (*entailed*) represents *entailed* and 0 (*not-entailed*) represents *not-entailed*. One can co-relate model predictions for related TE pairs with same context but different hypothesis during prediction (inference) to retrieve the classification label. This is depicted by an example in Table 4. We call our approach a *two-step classification* method, where we obtain TE predictions in the first step and use them to obtain classification label in step two. For demarcation, we refer to the straightforward task (without the recasted data) as *direct classification*.

Therefore, a perfect TE model would lead to a 100% accuracy over the *two-step classification* task. However, having a completely accurate TE model is often a bottleneck due to inaccurate and inconsistent predictions. Here, inconsistency can even occur across pairs, for example, two different pairs can predict two different labels. So instead of binary outputs, we use soft TE probabilities ( $p_i$ ) of each context-hypothesis pair ( $c_i-h_i$ ) and concatenate them together to form an *entailment vector* ( $\mathcal{E}$ ), see Figure 1. The classifier  $\mathcal{C} : \mathcal{E} \rightarrow$

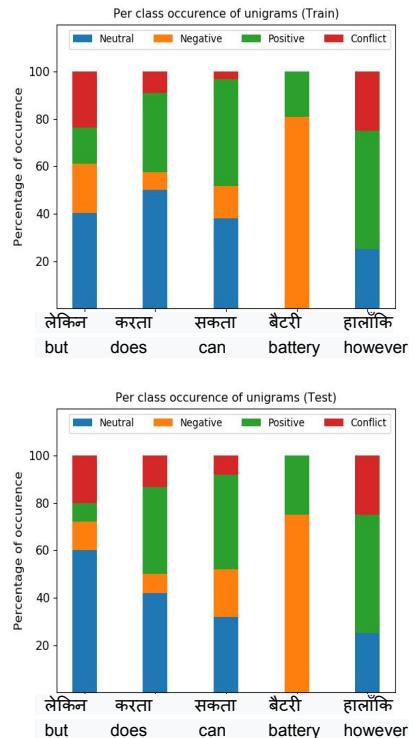


Figure 2: Plot showing statistics of unigram patterns in PR dataset for train (top) and test (bottom) across different classes for some sentiment as well as non-sentiment keywords. The x-axis represents the keyword with the percentage of occurrence on the y-axis.

$\mathcal{Y}$ , then takes as input the *entailment vector* ( $\mathcal{E}$ ) to retrieve the classification label ( $\mathcal{Y}$ ). Here, the *entailment vector* works as an added weaker supervision at the group level (group of all recasted pairs for a given context) to the classifier. Thus the classifier identify the correct boundary for the final classification task.

Furthermore, *two-step classification* adds an interpretable advantage over the direct classification. This is because, direct-classification is driven by a lot of spurious unigram patterns present in the original dataset. These patterns are leveled in the *two-step classification* approach due to the balanced set of text tokens

for both entailed and not-entailed pairs (both labels) with data recasting. Figure 2 shows some of the unigram statistics for *PR* dataset over some sentiment as well as non-sentiment words to depict the type of artefact patterns in the classification datasets, similar to (Tan et al., 2019). These annotation artefacts are nullified in the recasted TE task due to balanced label balanced for every premise tokens.

#### 4.2.1 Joint Objective (JO)

One simple method for *two-step classification* is to first train a TE model and then train the classifier on its predictions. However, using a fixed TE model prediction imposes a prior bottleneck on the classification accuracy. Since both the tasks i.e. the TE and the follow-up classification, can influence each other, thus we propose a joint training objective as shown in Equation 2

$$\mathcal{L}_{joint} = \mathcal{L}_{TE} + \lambda \mathcal{L}_{clf} \quad (2)$$

where  $\lambda$  is the weight of the follow-up classification loss,  $\mathcal{L}_{TE}$  and  $\mathcal{L}_{clf}$  are cross-entropy loss for the task of TE and classification respectively as defined in Equations 3 and 4.

$$\mathcal{L}_{TE} = \sum_k \sum_{j=1}^m -p_{k,j}^{true} \log p_{k,j} \quad (3)$$

$$\mathcal{L}_{clf} = \sum_k \sum_{j=1}^m -c_{k,j}^{true} \log c_{k,j} \quad (4)$$

Here,  $m$  represents the total classes,  $p_{k,j}^{true}$  and  $c_{k,j}^{true}$  represent the binary label of sample  $k$  to belong to class  $j$ , and  $p_{k,j}$  and  $c_{k,j}$  represent the probability of predicted label for sample  $k$  to be class  $j$ .

**Benefit of Joint Objective.** Satisfying the joint objective not only ensures that the model predictions are correct but also ensures that they are correct for the right reasons. The true classification label can be retrieved from the entailment vector only when the model draws necessary inferences correctly. Otherwise the multi-class classification would fail. Furthermore, combining the joint objective (Equation 2) with consistency regulariser (Equation 1) for the intermediate TE prediction further force pairwise-consistency between prediction of related TE pairs.

Context sentence: He cried over his lost pet.	
Hypotheses	TE Prediction
1. He is happy.	<i>not-entailed</i>
2. He is not happy.	<i>entailed</i>
3. He is angry.	<i>not-entailed</i>
4. He is not angry.	<i>entailed</i>
5. He is sad.	<i>entailed</i>
6. He is not sad.	<i>not-entailed</i>
Inferred label: Sad	

Table 4: An example demonstrating inference of the label for the original classification task based on predictions from TE model.

## 5 Experiments

Most of the sentence embedding models have been designed and evaluated to perform well on *English* language. The experiments in this work are motivated to answer the following questions for a low-resource language, *Hindi*:

- Are these representations effective to derive logical entailment in context-hypothesis pairs on recasted data?. Furthermore, how consistent/inconsistent are such models with their own decisions? Also, does consistency regulariser help to mitigate model inconsistency?
- Do sentence representation models work well for direct classification? Can models trained on recasted NLI data be used to retrieve ground truth classification annotations using *two-step classification*? Does our joint training objective with consistency regularization improve performance?

**Baselines** - For evaluating our approach, we use the following baselines: InferSent (Conneau et al., 2017), Sent2Vec (Pagliardini et al., 2018), Bag-of-words (BoW) and XLM-RoBERTa (Conneau et al., 2019) which is state-of-the-art for multilingual language modelling. Also, we evaluate a hypothesis-only analogue for each one of them as well. For experiments with recasted data, we use embeddings of *context-hypothesis* pair for baselines whereas for the hypothesis-only (Poliak et al., 2018b) models, we only use embeddings of the *hypothesis* sentence, keeping it blind to the *context*.

**Hypothesis only Baselines** - Evaluating hypothesis-only models is motivated by irregularities and biases presented in entailment

Context (Hindi): वह रोया जब उसने अपना पालतू खो दिया		Emotion class (Hindi): दुख (English): Sad		
Hypothesis (Hindi)	Hypothesis (English)	TE label	Consistency	Prediction
$h1 : \text{वह खुश है}$	$h1 : \text{He is happy.}$	<i>not-entailed</i>	Consistent	Correct
$h1' : \text{वह खुश नहीं है}$	$h1' : \text{He is not happy.}$	<i>entailed</i>	Consistent	Correct
$h1 : \text{वह खुश है}$	$h1 : \text{He is happy.}$	<i>not-entailed</i>	Inconsistent	Correct
$h1' : \text{वह खुश नहीं है}$	$h1' : \text{He is not happy.}$	<i>not-entailed</i>	Inconsistent	Incorrect
$h1 : \text{वह खुश है}$	$h1 : \text{He is happy.}$	<i>entailed</i>	Inconsistent	Incorrect
$h1' : \text{वह खुश नहीं है}$	$h1' : \text{He is not happy.}$	<i>entailed</i>	Inconsistent	Correct
$h1 : \text{वह खुश है}$	$h1 : \text{He is happy.}$	<i>entailed</i>	Consistent	Incorrect
$h1' : \text{वह खुश नहीं है}$	$h1' : \text{He is not happy.}$	<i>not-entailed</i>	Consistent	Incorrect

Table 5: A simple example illustrating the concept of consistency in model prediction for TE task for the task of emotion analysis.

datasets. Such biases often lead to high performance over NLI tasks without completely comprehending the semantic reasonings in data and language. When the accuracy of a hypothesis-only model is much lower than the baseline and closer to random (50%), it exhibits that learning is not boosted due to statistical irregularities in data such as word count, unigram/bi-gram pattern or any other spurious pattern (artefacts). We achieve this using our approach since recasting ensures label balance for the augmentations of each class label for every sentence and its tokens.

**Experimental Settings** - For each of the models, we use the initial learning rate  $1 \times 10^{-3}$  and a decay rate of 0.9, using Adam optimizer with the embedding dimension kept as 1024 for all the models. For all the experiments associated with XLM-RoBERTa, We use XLM-RoBERTa large with 1024-hidden. For InferSent and Sent2Vec we use the default parameter for NLI model architecture as stated in the paper. For hypothesis only baseline we use the single sent model of XLM-RoBERTa, InferSent and Sent2Vec as reported in paper for binary classification.

After the embeddings are obtained, we use an MLP classifier for performing all the classification experiments. For a hypothesis-only baseline, only the hypothesis embedding is passed as an input to the MLP, whereas for a premise-hypothesis baseline, we concatenate the embeddings of premise, hypothesis, as well as their element-wise product and element-wise subtraction. For the joint objective training (see Eq. 2), we use  $\lambda=2.0$ . We train our model for 15 epochs on a machine with GeForce RTX 2080 GPU using the PyTorch framework.

## 5.1 Textual Entailment Results

For all four semantic phenomenon considered, we use recasted data to predict the performance on textual entailment task. While training, we use four *context-hypothesis* pairs - with hypothesis having true classification label, its negation (hypothesis 5 and 6 in Table 4), a random label from the remaining classes and its negation (hypothesis 1 and 2 in Table 4). This ensures that neither original classification label nor the negation (we choose only one random pair) correlate with entailment labels. For development and test sets, we use all possible  $2n$  recasted pairs (where  $n$  is the number of classes in classification data) since ideally, while testing we have no prior knowledge of the ground-truth label.

Context-Hypothesis Baselines				
Sentence Representation	Dataset			
	PR	BH	HDA	BBC
BoW	47.32	51.00	54.20	57.00
Sent2Vec	61.21	62.67	64.00	65.42
InferSent	68.00	65.04	67.9	68.84
XLM-RoBERTa	<b>74.02</b>	<b>74.48</b>	<b>75.29</b>	<b>73.56</b>
Hypothesis-only Baselines				
BoW	44.89	47.01	44.82	43.00
	51.91	50.84	50.88	48.80
Sent2Vec	54.32	52.14	53.54	51.08
	<b>55.00</b>	<b>52.60</b>	<b>53.92</b>	<b>55.00</b>

Table 6: TE classification accuracies using different sentence embeddings for all four datasets.

With Table 6, we establish that XLM-RoBERTa (Conneau et al., 2019) gives the best performance as compared to all the other baselines. Therefore, we use it for all the following experiments. Also, random performance on hypothesis-only baseline ensures that our recasted data does not contain hypothesis-bias.

**Consistency** - We analyse the effect of consistency regulariser (CR) by comparing the percentage of inconsistent model predictions for TE models with and without CR. Figure 3 clearly depicts that the constraint regularisation helps in reducing the percentage of inconsistent pairs and hence makes the model predictions congruent with its own internal representation in the model parameters.

## 5.2 Two-step Classification Results

We now use the TE model to perform *two-step classification* as explained in section 4.2. Table 10 shows the classification accuracies obtained via direct as well as *two-step* classification with consistency regularisation and joint-objective. As reported in Table 9 and 10, we observe a jump in both the TE as well as *two-step classification* accuracies with the addition of consistency regularisation. Such a constraint restricts the model predictions to be either correct or incorrect but not pairwise-inconsistent with its other beliefs.

**Joint Objective** - In Table 9 and 10, we observe that joint objective proves to be much more beneficial than independent TE and classifier training. The *two-step classification* accuracy with joint-objective (+JO+CR) surpasses the direct classification performance.

We observe an increment of 5% in TE and 2% in classification accuracy across all the datasets. Furthermore, from Figure 3, we observe that, JO also improve the prediction consistency across all the datasets. Table 7 shows the exact percentage of correct/incorrect and inconsistent pairs.

**Improved Performance Analysis** - The two-step classification is able to achieve overall improvement over direct classification approach mainly due to following two factors. Firstly, the joint objective (JO) helps in creating a feedback loop with the two tasks of textual entailment and classification, which enforce consistency in the model predictions for the two tasks. Secondly, the consistency regularisation (CR) for the TE helps in making the model decisions congruent across same context premise but different related hypothesis. Thus, both the JO and CR imposes indirect and direct inductive bias through constrained loss objective which improves model performance

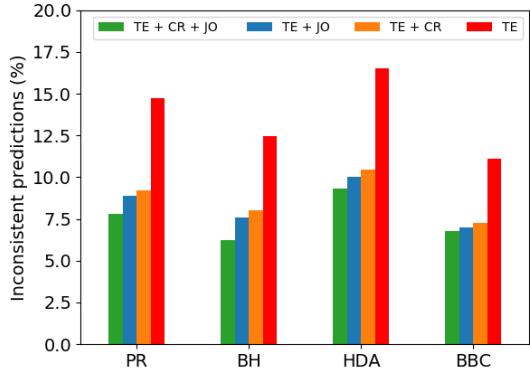


Figure 3: Plot depicting percentage (%) of inconsistent predictions for all the datasets using *XLM-RoBERTa* with and without consistency regularisation (CR) and Joint Objective (JO).

compared to the direct classification task.

## 5.3 Direct vs Two-Step Classification

We analyse the classification predictions obtained by direct as well as two-step classification to compare the differences. Figure 4 shows the percentage (%) of correct and incorrect predictions obtained for the two approaches considered. More generally, we see a maximum consensus across the main diagonal between the two approaches. However, there are irregularities wherein one of the predictions contradicts the other.

As illustrated in Table 8, we depict qualitative examples corresponding to these irregularities. We analyse their entailment vectors to interpret intermediate predictions and realise that the high entailments corresponding to the gold label and certain incorrect label lead to incorrect predictions. For example, for the first sentence in Table 8, we observe that the context-hypothesis pairs with hypothesis corresponding to *The product received negative reviews from its users*, and ‘*The product received conflicting reviews from its users*’ get the entailment probabilities 0.64 and 0.58, respectively. This shows that apart from the gold label i.e. *negative* here, there is an inclination towards the class label *conflict*.

Moreover, we see certain statistical word patterns like the usage of the keyword *but* in most of the sentences corresponding to the class *conflict*, thereby ensuring a certain degree of artefact learning which governs the decisions in direct classification. One advan-

Dataset	Correct				Incorrect				Inconsistent			
	TE	+CR	+JO	+CR +JO	TE	+CR	+JO	+CR +JO	TE	+CR	+JO	+CR +JO
PR	71.43	72.18	72.50	<b>74.00</b>	<b>13.82</b>	18.6	18.6	18.2	14.75	9.22	8.90	<b>7.80</b>
BH	73.20	74.50	74.76	<b>75.80</b>	<b>14.32</b>	17.50	17.66	17.99	12.48	8.00	7.58	<b>6.21</b>
HDA	72.00	74.88	75.22	<b>76.8</b>	<b>11.50</b>	14.66	14.78	13.9	16.50	10.46	10.00	<b>9.30</b>
BBC	71.17	74.56	74.84	<b>76.00</b>	<b>17.75</b>	18.2	18.16	17.2	11.08	7.24	7.00	<b>6.80</b>

Table 7: Percentage (%) of correct, incorrect and inconsistent prediction pairs for all the datasets using XLM-RoBERTa.

Sentence	True Label	Direct clf.	Two-step clf.
यहाँ खाना पीना उतना मँहगा नहीं पर रहना जेब को काफी भारी पड़ता है। <b>English:</b> Drinking here is not that expensive but living on the pocket is very heavy.	<i>negative</i>	<i>conflict</i>	<i>negative</i>
राजगुरु , महाराज कृष्णदेव राय को कहते हैं के तेनालीराम झूठ बोल रहे हैं। <b>English:</b> Rajguru tells Maharaja Krishnadeva Raya that Tenaliram is lying.	<i>anger</i>	<i>anger</i>	<i>sad</i>

Table 8: Qualitative examples where direct and two-step classification methods contradict predictions.

Dataset	Textual Entailment			
	w/o CR/JO	+CR	+JO	+CR+JO
PR	74.02	77.80	78.40	<b>81.40</b>
BH	74.48	76.57	77.01	<b>80.05</b>
HDA	75.29	78.00	78.22	<b>81.67</b>
BBC	73.56	76.24	77.69	<b>79.22</b>

Table 9: TE accuracies for all the four datasets using XLM-RoBERTa (Conneau et al., 2019).

Dataset	Direct clf.	Two-step clf.			
		TE	TE+ CR	TE+ JO	TE+ CR+JO
PR	71.65	66.24	69.38	70.58	<b>73.70</b>
BH	73.03	68.06	70.91	71.82	<b>74.80</b>
HDA	74.25	68.22	71.45	72.45	<b>75.96</b>
BBC	70.22	65.98	68.20	70.30	<b>72.18</b>

Table 10: Classification (direct and two-step) accuracies for all the four datasets using XLM-RoBERTa (Conneau et al., 2019).

tage of two-step classification is that it is more transparent about its predictions. This ensures more interpretability in the model decisions. We also compare class-wise accuracies of both the approaches for each of the datasets and see improvements with the two-step method in all classes<sup>9</sup>.

## 6 Conclusion

In this work, we share the first recasted NLI dataset in a low-resource language Hindi, and show how a large-scale NLI data can be developed for low-resource languages without un-

<sup>9</sup>See Appendix Section A.2 Figure 6 for class-wise results

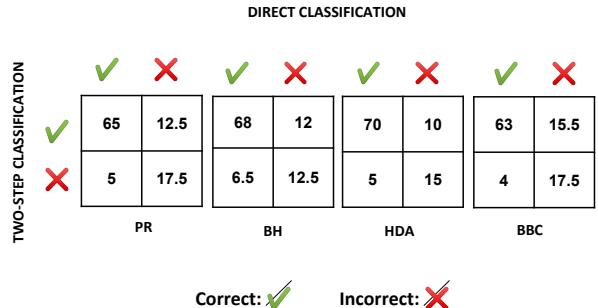


Figure 4: Correct vs Incorrect Predictions (%) for Direct and Two-Step classification.

dergoing costly and time taking human annotations. We perform TE experiments and introduce a consistency regulariser to avoid pairwise-inconsistent TE predictions. Furthermore, we propose a *two-step* classification approach with a joint training objective. Our results with the joint objective shows significant improvement in performance.

As a future work, we aim to analyse the proposed methodology which is language independent on other low-resource languages. We also aim to use more generalisable templates for linguistic diversity in recating data. It would be interesting to analyse how extending textual entailment knowledge especially the consistency regularization constraint affect other downstream NLP tasks apart from textual classification, not only in terms of the performance, but also in enhancing the model interpretability.

## References

- Md Shad Akhtar, Ayush Kumar, Asif Ekbal, and Pushpak Bhattacharyya. 2016. A hybrid deep learning architecture for sentiment analysis. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 482–493.
- Pushpak Bhattacharyya. 2012. Natural language processing: A perspective from computation in presence of ambiguity, resource constraint and multilinguality. *CSI journal of computing*, 1(2):1–13.
- Samuel Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642.
- Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, F. Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *ArXiv*, abs/1911.02116.
- Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrau, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680.
- Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485.
- Swapnil Dhanwal, Hritwik Dutta, Hitesh Nankani, Nilay Shrivastava, Yaman Kumar, Junyi Jessy Li, Debanjan Mahata, Rakesh Gosangi, Haimin Zhang, Rajiv Ratn Shah, and Amanda Stent. 2020. An annotated dataset of discourse modes in Hindi stories. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 1191–1196, Marseille, France. European Language Resources Association.
- Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalkika Bali, and Monojit Choudhury. 2020. The state and fate of linguistic diversity and inclusion in the NLP world. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online. Association for Computational Linguistics.
- Ramchandra Joshi, Purvi Goel, and Raviraj Joshi. 2019. Deep learning for hindi text classification: A comparison. In *International Conference on Intelligent Human Computer Interaction*, pages 94–101. Springer.
- Simran Khanuja, S. Dandapat, S. Sitaram, and M. Choudhury. 2020. A new dataset for natural language inference from code-mixed conversations. In *CodeSwitch@LREC*.
- Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. Scitail: A textual entailment dataset from science question answering. In *Thirty-Second AAAI Conference on Artificial Intelligence*.
- Yaman Kumar, Debanjan Mahata, Sagar Aggarwal, Anmol Chugh, Rajat Maheshwari, and Rajiv Ratn Shah. 2019. Bhaav- a text corpus for emotion analysis from hindi stories. *ArXiv*, abs/1910.04073.
- Guanyu Li, Pengfei Zhang, and Caiyan Jia. 2018. Attention boosted sequential inference model. *CoRR*, abs/1812.01840.
- Tao Li, Vivek Gupta, Maitrey Mehta, and Vivek Srikanth. 2019. A logic-driven framework for consistency of neural models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*.
- Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4496.
- Puneet Mathur, Rajiv Shah, Ramit Sawhney, and Debanjan Mahata. 2018. Detecting offensive tweets in hindi-english code-switched language. In *Proceedings of the Sixth International Workshop on Natural Language Processing for Social Media*, pages 18–26.
- Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. 2018. Unsupervised learning of sentence embeddings using compositional n-gram features. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 528–540.
- Adam Poliak, Aparajita Halder, Rachel Rudinger, J. Edward Hu, Ellie Pavlick, Aaron Steven White, and Benjamin Van Durme. 2018a. Collecting diverse natural language inference problems for sentence representation evaluation. In *BlackboxNLP@EMNLP*.
- Adam Poliak, Jason Naradowsky, Aparajita Halder, Rachel Rudinger, and Benjamin Van Durme. 2018b. Hypothesis only baselines in natural language inference. In *\*SEM@NAACL-HLT*.

Mark Sammons, V.G.Vinod Vydiswaran, and Dan Roth. 2010. “ask not what textual entailment can do for you...”. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1199–1208, Uppsala, Sweden. Association for Computational Linguistics.

Kushagra Singh, Indira Sen, and Ponnurangam Kumaraguru. 2018. Language identification and named entity recognition in hinglish code mixed tweets. In *Proceedings of ACL 2018, Student Research Workshop*, pages 52–58.

Sunayana Sitaram, Khyathi Raghavi Chandu, Sai Krishna Rallabandi, and Alan W Black. 2019. A survey of code-switched speech and language processing. *arXiv preprint arXiv:1904.00784*.

Shawn Tan, Yikang Shen, Chin-Wei Huang, and Aaron C. Courville. 2019. Investigating biases in textual entailment datasets. *ArXiv*, abs/1906.09635.

Marta Tatu and Dan Moldovan. 2005. A semantic approach to recognizing textual entailment. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 371–378, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Rui Wang and Yi Zhang. 2009. Recognizing textual relatedness with predicate-argument structures. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*, pages 784–792. Association for Computational Linguistics.

Aaron Steven White, Pushpendre Rastogi, Kevin Duh, and Benjamin Van Durme. 2017. Inference is everything: Recasting semantic resources into a unified evaluation framework. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 996–1005.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

## A Appendix

### A.1 Illustration of Recasting Approach

We illustrate the proposed recasting approach in more detail with example templates in Fig-

ure 5. We show how each classification sentence is used to create a context-hypothesis pair for NLI task for different datasets corresponding to the diverse semantic phenomenon considered.

## A.2 Additional Results

**Development Set Results** - We report the results on development set for textual entailment as well as classification in Table 12 and 13 respectively. We observe similar trends in the development set as depicted in the test set performance for both the tasks of textual entailment as well as the *two-step* classification task.

**Class-wise Performance** - In Figure 6, we show class-wise accuracies obtained by the two classification approaches - direct vs two-step. Broadly, we obtain a considerable improvement in the performance of two-step classification over direct classification, over all classes across all the four datasets. This ensures that the obtained performance improvement is balanced across all classes.

**Semi-supervised setting** - We extend our analysis to a semi-supervised setting (with fewer labels) wherein we retain the true labels for only 40%, 60% and 80% of the data while training and analyse its effect on the performance of TE and classification tasks.

Table 14, 16 and 18 show the results obtained with different ablations with 80%, 60% and 40% of the labelled data respectively for the TE task. Similarly, Table 15, 17 and 19 report the results for direct and two-step classification in the semi-supervised approach highlighting the effect of joint objective and consistency regularisation in obtaining improvement.

Although, we utilize the consistency regularisation, since it does not depend on the true label, rather operated on pairwise context-hypothesis groupings. We observe that TE with consistency regularisation and joint objective surpasses the trivial TE task without any added constraints. This depicts that our regularisation and joint objective approach add robust improvements in TE model performance even with minimum supervision.

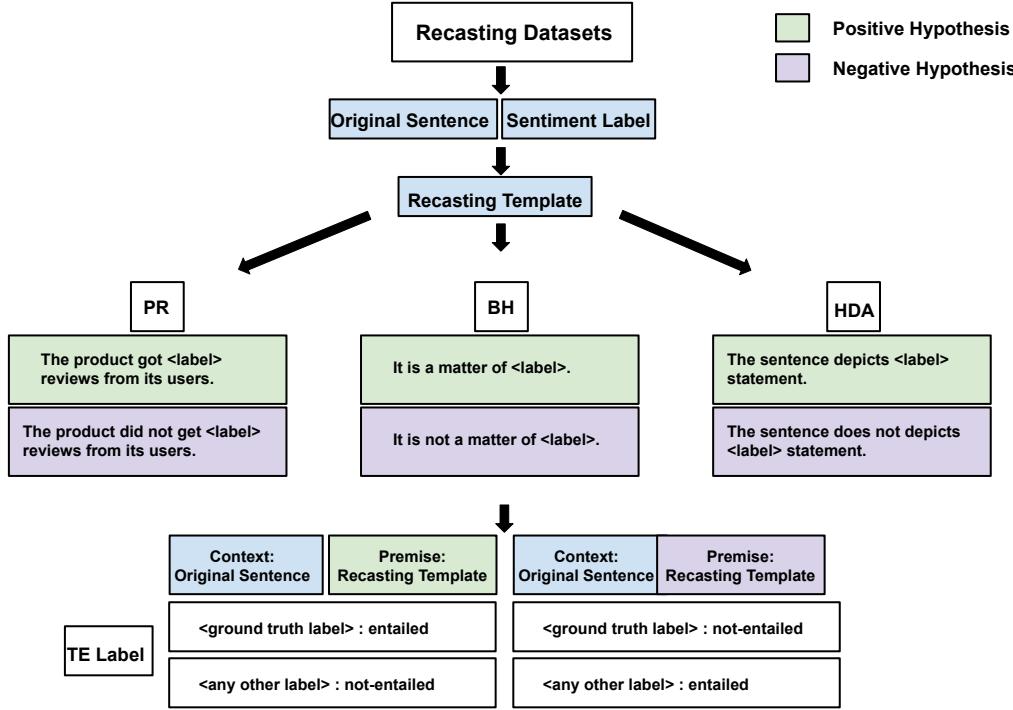


Figure 5: Illustration of the proposed recasting approach.

Original Sentence(Hindi)	Original Sentence (English)	Sentiment	
इन पवित्र भार्वों से उसकी आत्मा विहँल हो गयी।	His soul was overwhelmed by these holy feelings.	Joy	
<b>Model Consistency/Inconsistency</b>			
Contradictory TE pairs (Hindi)	Contradictory TE pairs (English)	Prediction	Label
<i>p:</i> इन पवित्र भार्वों से उसकी आत्मा विहँल हो गयी। <i>h1:</i> क्या यह खुशी की बात है?	<i>p:</i> His soul was overwhelmed by these holy feelings. <i>h1:</i> Is this a matter of joy?	<i>e</i>	Inconsistent
<i>p:</i> इन पवित्र भार्वों से उसकी आत्मा विहँल हो गयी। <i>h2:</i> क्या यह खुशी की बात नहीं है?	<i>p:</i> His soul was overwhelmed by these holy feelings. <i>h2:</i> Is this not a matter of joy?	<i>e</i>	Correct
		<i>ne</i>	Incorrect
		<i>ne</i>	Inconsistent

Table 11: Example sentences for contradictory premise (*p*) - (*h*) pairs for measuring inconsistency in the recasted model predictions with *e* representing *entailed* and *ne* representing *not-entailed*.

Dataset	Textual Entailment ↑			
	w/o	+CR	+JO	+CR+JO
PR	74.26	78.44	78.02	<b>80.60</b>
BH	73.88	76.46	76.82	<b>80.95</b>
HDA	75.90	78.54	78.48	<b>81.86</b>
BBC	73.45	76.48	77.96	<b>79.02</b>

Table 12: TE accuracies for all the four datasets using XLM-RoBERTa on the development set.

Dataset	Direct clf.	Two-step clf. ↑			
		TE	TE+CR	TE+JO	TE+CR+JO
PR	71.40	65.48	68.76	70.84	<b>72.98</b>
BH	73.50	69.24	70.88	71.46	<b>75.66</b>
HDA	74.85	68.46	72.34	73.50	<b>75.56</b>
BBC	71.36	66.40	68.38	70.47	<b>73.08</b>

Table 13: Classification (direct and two-step) accuracies for all the four datasets using XLM-RoBERTa on the development set.

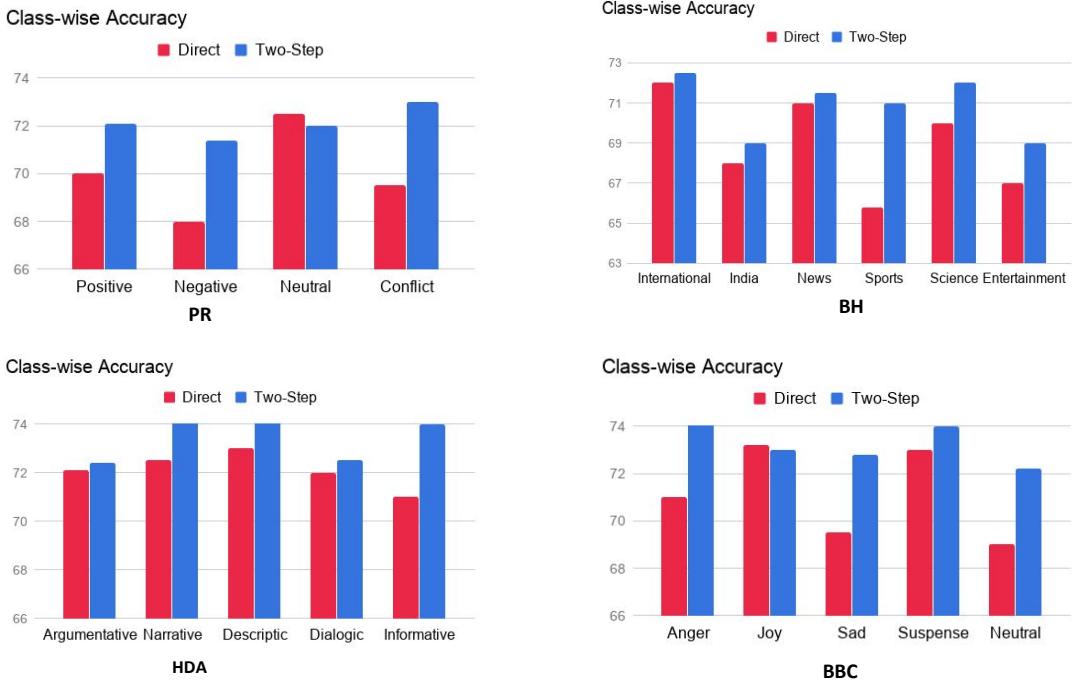


Figure 6: Class-wise comparison of Direct vs Two-Step Classification.

Dataset	Textual Entailment ↑			
	w/o	+CR	+JO	+CR+JO
PR	69.23	72.68	70.48	<b>74.04</b>
BH	70.65	71.09	70.99	<b>73.98</b>
HDA	70.29	72.23	71.32	<b>74.67</b>
BBC	70.36	73.84	71.65	<b>74.52</b>

Table 14: TE accuracies for all the four datasets using XLM-RoBERTa with fewer labels (80%).

Dataset	Direct clf.	Two-step clf. ↑			
		TE	TE+CR	TE+JO	TE+CR+JO
PR	67.20	61.28	64.87	62.49	<b>68.98</b>
BH	68.51	64.22	66.71	71.46	<b>69.46</b>
HDA	68.82	62.62	65.13	63.75	<b>69.95</b>
BBC	66.93	60.94	63.14	61.47	<b>67.73</b>

Table 15: Classification (direct and two-step) accuracies for all the four datasets using XLM-RoBERTa with fewer labels (80%).

Dataset	Textual Entailment ↑			
	w/o	+CR	+JO	+CR+JO
PR	65.12	67.46	65.58	<b>70.06</b>
BH	66.12	68.57	67.22	<b>70.69</b>
HDA	65.29	67.25	66.34	<b>70.59</b>
BBC	66.87	68.22	67.19	<b>71.42</b>

Table 16: TE accuracies for all the four datasets using XLM-RoBERTa with fewer labels (60%).

Dataset	Direct clf.	Two-step clf. ↑			
		TE	TE+CR	TE+JO	TE+CR+JO
PR	60.29	61.82	62.37	62.00	<b>63.98</b>
BH	61.52	62.14	64.18	62.45	<b>64.81</b>
HDA	61.82	63.47	63.94	63.33	<b>65.56</b>
BBC	60.23	61.24	62.16	62.09	<b>64.73</b>

Table 17: Classification (direct and two-step) accuracies for all the four datasets using XLM-RoBERTa with fewer labels (60%).

Dataset	Textual Entailment ↑			
	w/o	+CR	+JO	+CR+JO
PR	57.12	58.46	58.08	<b>59.56</b>
BH	59.12	59.57	59.22	<b>60.69</b>
HDA	59.29	59.25	60.19	<b>60.78</b>
BBC	58.42	58.70	58.10	<b>59.02</b>

Table 18: TE accuracies for all the four datasets using XLM-RoBERTa with fewer labels (40%).

Dataset	Direct clf.	Two-step clf. ↑			
		TE	TE+CR	TE+JO	TE+CR+JO
PR	55.29	56.28	56.48	57.00	<b>59.89</b>
BH	58.52	59.17	59.18	59.59	<b>60.11</b>
HDA	58.82	58.43	58.94	59.23	<b>60.68</b>
BBC	55.23	57.24	56.46	58.01	<b>60.78</b>

Table 19: Classification (direct and two-step) accuracies for all the four datasets using XLM-RoBERTa with fewer labels (40%).

### A.3 Another Inconsistency Example

In Table 11, we explain the concept of pairwise consistencies and inconsistencies in the context-hypothesis pairs in the recasted data with an example. It depicts how different entailment results for the same context but different hypothesis can lead to inconsistencies within the model predictions.

### A.4 Benefits of Data Recasting

There are several benefits of data recasting (Conneau et al., 2019) especially for low-resource languages

- Recasting is an automated process and hence remove the need of expensive human annotation to labelled data.
- Uniform procedure of recasting data has equal number of *context-hypothesis* pairs for each label, hence making it neutral to statistical irregularities (see hypothesis bias experiments in Section 5).
- Diverse semantic phenomenon for various classification tasks can be unified as a single task using data recasting.

# Explaining Word Embeddings via Disentangled Representation

Keng-Te Liao

National Taiwan University  
d05922001@ntu.edu.tw

Zhong-Yu Huang

National Taiwan University  
r06944047@ntu.edu.tw

Cheng-Syuan Lee

National Taiwan University  
r07922055@ntu.edu.tw

Shou-de Lin

National Taiwan University  
sdlin@csie.ntu.edu.tw

## Abstract

Disentangled representations have attracted increasing attention recently. However, how to transfer the desired properties of disentanglement to word representations is unclear. In this work, we propose to transform typical dense word vectors into disentangled embeddings featuring improved interpretability via encoding polysemous semantics separately. We also found the modular structure of our disentangled word embeddings helps generate more efficient and effective features for natural language processing tasks.

## 1 Introduction

Disentangled representations are known to represent interpretable factors in separated dimensions. This property can potentially help people understand or discover knowledge in the embeddings. In natural language processing (NLP), works of disentangled representations have shown notable impacts on sentence and document-level applications. For example, Larsson et al. (2017) and Melnyk et al. (2017) proposed to disentangle sentiment and semantic of sentences. By manipulating sentiment factors, the machine can rewrite a sentence with different sentiment. Brunner et al. (2018) also demonstrated sentence generation while more focusing on syntactic factors such as part-of-speech tags. For document-level applications, Jain et al. (2018) presented a learning algorithm which embeds biomedical abstracts disentangling populations, interventions and outcomes. Regarding word-level disentanglement, Athiwaratkun and Wilson (2017) proposed mixture of Gaussian models which can disentangle meanings of polysemous words into two or three clusters. It has a connection with unsupervised sense representations (Camacho-Collados and Pilehvar, 2018) which is an active research topic in the community.

In this work, we focus on word-level disentanglement and introduce an idea of transforming dense word embeddings such as GloVe (Pennington et al., 2014) or word2vec (Mikolov et al., 2013b) into disentangled word embeddings (DWE). The main feature of our DWE is that it can be segmented into multiple sub-embeddings or sub-areas as illustrated in Figure 1. In the figure, each sub-area encodes information relevant to one specific topical factor such as *Animal* or *Location*. As an example, we found words similar to “turkey” are “geese”, “flock” and “goose” in the *Animal* area, and the similar words turn into “Greece”, “Cyprus” and “Ankara” in the *Location* area.

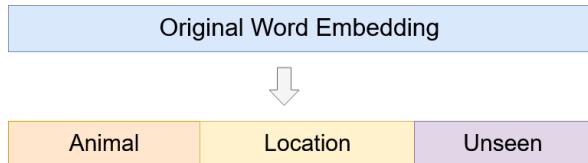


Figure 1: Disentangled embedding with factors *Animal*, *Location* and *Unseen*.

We also found our DWE generally satisfies the *Modularity* and *Compactness* properties proposed by Higgins et al. (2018) and Ridgeway and Mozer (2018) which can be a definition of general-purpose disentangled representations. Also, our DWE can have the following advantages:

- **Explaining Underlying Knowledge**

The multi-senses of words can be extracted and separately encoded despite the learning algorithm of the original word embeddings (e.g. GloVe) does not do disambiguation. As a result, the encoded semantic can be presented in an intuitive way for examination.

- **Modular and Compact Features**

Each sub-area of our DWE can itself be informative features. The advantage is that people

are free to abandon features in sub-areas irrelevant to the given downstream tasks while still achieving competitive performance. In Section 4, we show that using the compact features is not only efficient but also helps improve performance on downstream tasks.

- **Quality Preservation**

In addition to higher interpretability, our DWE preserves co-occurrence statistics information in the original word embeddings. We found it also helps preserve the performance on downstream tasks including word similarity, word analogy, POS-tagging, chunking, and named entity recognition.

## 2 Obtaining Disentangled Word Representations

### 2.1 Problem Definition

Our goal is transforming  $N d$ -dimensional dense word vectors  $X \in \mathbb{R}^{N \times d}$  into disentangled embeddings  $Z \in \mathbb{R}^{N \times d}$  by leveraging a set of binary attributes  $\mathcal{A} = \{a_1, \dots, a_M\}$  labelled on words.

$Z$  is expected to have two properties. The first one is preserving word features encoded in  $X$ . More specifically, we require  $XX^T \approx ZZ^T$  as pointed out by Levy and Goldberg (2014) that typical dense word embeddings can be regarded as factorizing co-occurrence statistics matrices.

The second property is that  $Z$  can be decomposed into  $M+1$  sub-embedding sets  $Z_{a_1}, \dots, Z_{a_M}$  and  $Z_{unseen}$ , where each sub-embedding set encodes information only relevant to the corresponding attribute. For example,  $Z_{a_1}$  is expected to be relevant to  $a_1$  and irrelevant to  $a_2, \dots, a_M$ . Information in  $X$  not relevant to any attributes in  $\mathcal{A}$  is then encoded in  $Z_{unseen}$ . An example of transforming  $X$  into  $Z$  with two attributes, *Animal* and *Location*, is illustrated in Figure 1.

For modelling the relevance between sub-embeddings and attributes, we use mutual information  $\mathcal{I}(Z_a, a)$  as learning objectives, where  $a$  is an arbitrary attribute in  $\mathcal{A}$ .

### 2.2 Transformation with Quality Preservation

We obtain  $Z$  by transforming  $X$  by a matrix  $W \in \mathbb{R}^{d \times d}$ . That is,  $Z = XW$ . To ensure  $XX^T \approx ZZ^T$ , an additional constraint  $WW^T = I$  is included.  $ZZ^T = (XW)(XW)^T = X(WW^T)X^T = XX^T$  if  $WW^T = I$  holds.

### 2.3 Optimizing $\mathcal{I}(Z_a, a)$

Let  $z_{a,i}$  be the  $i$ -th row in  $Z_a$ . By derivation,  $\mathcal{I}(Z_a, a) =$

$$\begin{aligned} & \sum_{i=1}^N p(z_i)p(a|z_{a,i}) [\log p(a|z_{a,i}) - \log p(a)] \\ & \approx \frac{1}{N} \sum_{i=1}^N p(a|z_{a,i}) [\log p(a|z_{a,i}) - \log p(a)] \end{aligned}$$

We let  $\log p(a)$  be constant and replace  $p(a|z)$  with a parametrized model  $q_\theta(a|z)$ . By experiments, we found logistic regression with parameter  $\theta$  is sufficient to be  $q_\theta(a|z)$ . Intuitively, high  $\mathcal{I}(Z_a, a)$  means  $Z_a$  are informative features for a classifier to distinguish whether words has attribute  $a$ .

When increasing  $\mathcal{I}(Z_a, a)$  by optimizing  $q_\theta(a|z)$ , we found a strategy helping generate higher quality  $Z$ . The strategy is letting  $Z_a$  be features to reconstruct original vectors for words having attribute  $a$ . For words with  $a$ , the approach becomes a semi-supervised learning architecture which attempts to predict labels and reconstruct inputs simultaneously.

The loss function  $\mathcal{L}(W, \theta, \phi)$  for maximizing  $\mathcal{I}(Z_a, a)$  is as follow:

$$\begin{aligned} & \frac{-1}{N} \sum_{i=1}^N q_\theta(a|z_{a,i}) + \lambda \mathbb{I}_{a,i} \|x_i - \phi(z_{a,i})\|_2^2 \\ & \mathbb{I}_{a,i} = \begin{cases} 1 & \text{when } i\text{-th word has attribute } a \\ 0 & \text{when } i\text{-th word does not have } a \end{cases} \end{aligned} \quad (1)$$

where  $\phi$  is single and fully-connected layer,  $x_i$  is the original  $i$ -th word's vector in  $X$ , and  $\lambda$  is a hyper-parameter. We set  $\lambda = \frac{1}{d}$  in all experiments.

### 2.4 Learning to Generate Sub-embedding $Z_a$

As discussed in 2.3 that high  $\mathcal{I}(Z_a, a)$  indicates  $Z_a$  are informative features for classification, we propose to regard sub-embedding generation as a feature selection problem. More specifically, we apply sparsity constraint on  $Z$ . Ideally, when predicting  $a$ , a smaller number of dimensions of  $Z$  are selected as the informative features, which are regarded as  $Z_a$ .

In this work, we use *Variational Dropout* (Kingma et al., 2015; Molchanov et al., 2017) as the sparsity constraint. At each iteration of training, a set of multiplicative noise  $\xi$  is sampled from a normal distribution  $\mathcal{N}(1, \alpha_a = \frac{p_a}{1-p_a})$  and injected on  $Z$ . That is, the prediction and reconstruction is done by  $\theta(\xi \odot Z)$  and  $\phi(\xi \odot Z)$ . The parameter  $\alpha_a \in \mathbb{R}^d$  is jointly learned with  $W$ ,  $\theta$ , and  $\phi$ . Afterwards,  $d$ -dimensional dropout rates

$p_a = \text{sigmoid}(\log \alpha_a)$  can be obtained. For each attribute  $a$  in  $\mathcal{A}$ , the dimensions with dropout rates lower than 50% are normally regarded as  $Z_a$ .

We would like to emphasize that the learned dropout rates are not binary values. Therefore, deciding the length of sub-embeddings can actually depend on users preferences or tasks requirements. For example, users can obtain more compact and pure  $Z_a$  by selecting dimensions with dropout rates lower than 10%, or get more thorough yet less disentangled  $Z_a$  by setting the threshold be 70%.

To encourage disentanglement when handling multiple attributes, we include additional loss functions on dropout rates. Let a  $M$ -dimensional vector  $P$  be  $1 - p_a$  for all  $a$  in  $\mathcal{A}$  in a specific dimension. The idea is to minimize  $\prod_{i=1}^M P_i$  with constraint  $\sum_{i=1}^M P_i = 1$ . The optimal solution is that the dimension is relevant to only one attribute  $a'$  where  $1 - p_{a'} \approx 1$ . In implementation, we minimize the following loss function

$$\sum_{i=1}^M \log P_i + \beta \|\sum_{i=1}^M P_i - 1\|_2^2 \quad (2)$$

We set  $\beta = 1$  in the experiments, and equation 1 and 2 are optimized jointly.

To generate  $Z_{unseen}$ , we initially select a set of dimensions and constrain their dropout rates be always larger than 50%. The number of dimensions of  $Z_{unseen}$  is a hyper-parameter. After selection, we do not apply equation 2 on the selected dimensions.

### 3 Evaluation

#### 3.1 Word Embeddings and Attributes

We transform 300-dimensional GloVe<sup>1</sup> into DWE. The 300-dimensional GloVe is denoted by GloVe-300. For word attributes  $\mathcal{A}$ , we use labels in **WordStat**<sup>2</sup>. WordStat contains 45 kinds of attributes labeled on 70,651 words. Among the attributes, we select 5 high-level and easily understandable attributes: *Artifact*, *Location*, *Animal*, *Adjective* (*ADJ*) and *Adverb* (*ADV*) for our experiments. The number of words labelled with these 5 attributes is 13,337. After training, all pre-trained GloVe vectors are transformed by the learned matrix  $W$  (i.e.  $XW$ ) for downstream evaluations.

The number of learned dimensions for each attribute is illustrated in Figure 2, where the threshold of dropout rates for dimension selection is 50%.

<sup>1</sup><https://nlp.stanford.edu/projects/glove/>

<sup>2</sup><https://provalisresearch.com/products/content-analysis-software/>

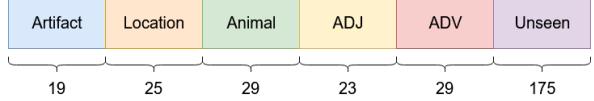


Figure 2: Disentangled embedding with five attributes: *Artifact*, *Location*, *Animal*, *Adjective* and *Adverb*. The remaining dimensions are viewed as *Unseen*.

	MEN	SimLex	BATS	GA
GloVe-300	0.749	0.369	<b>18.83</b>	<b>63.58</b>
DWE	<b>0.764</b>	<b>0.390</b>	18.75	62.30

Table 1: Word similarity and analogy performance.

	POS	Chunking	NER
GloVe-300	65.0	64.9	65.2
DWE	<b>67.2</b>	<b>66.3</b>	<b>66.1</b>

Table 2: POS-tag, chunking and NER performance.

#### 3.2 Evaluation of Quality Preservation

We firstly examine whether DWE can preserve features encoded in GloVe-300. The examination is done by intrinsic evaluations including the following tasks and datasets.

- Word Similarity: **Marco, Elia and Nam (MEN)** (Bruni et al., 2014) and **SimLex-999** (Hill et al., 2015).
- Word Analogy: **Bigger Analogy Test Set (BATS)** (Gladkova et al., 2016), **Google Analogy (GA)** (Mikolov et al., 2013a).
- POS tagging, Chunking and Named Entity Recognition (NER): **CoNLL 2003** (Sang and Meulder, 2003; Li et al., 2017).
- QVEC-CCA<sup>3</sup> (Tsvetkov et al., 2015): The performance is measured by semantic and syntactic CCA.

As shown in Table 1, 2 and 3, DWE can preserve performance of GloVe-300 on various NLP tasks. Probably due to the additional information of word attributes, DWE can have slightly better performance than GloVe-300 on seven of the tasks

#### 3.3 Attribute Classification

We design an attribute classification task for examining whether the DWE can meet requirements described in Section 2.3. We use logistic regression and take sub-embeddings  $Z_a$  as input features for

<sup>3</sup><https://github.com/ytsvetko/qvec>

	Semantic	Syntactic
GloVe-300	0.473	0.341
DWE	<b>0.474</b>	<b>0.348</b>

Table 3: QVEC-CCA evaluation.

	Artifact	Location	Animal	ADJ	ADV
$Z_{artifact}$	<b>77.8</b>	71.0	68.0	65.5	71.2
$Z_{location}$	59.2	<b>83.8</b>	64.0	60.5	69.8
$Z_{animal}$	58.5	67.5	<b>84.2</b>	60.2	71.0
$Z_{adj}$	69.8	70.7	68.2	<b>82.0</b>	72.5
$Z_{adv}$	59.0	72.5	71.8	71.5	<b>84.2</b>
$Z_{unseen}$	54.8	70.0	66.5	60.2	68.8

Table 4: Attribute classification accuracies (%).

verifying the performance of classification by cross-validation. For each attribute, We randomly sample 400 data for testing. The numbers of positive and negative data for testing are balanced. Therefore, a random predictor would get around 50% accuracy in each classification task.

The binary classification accuracies are shown in Table 4. Take the second column of Table 4 for example. For distinguishing whether a word can be location, taking  $Z_{location}$  as features for training a classifier achieves the highest accuracy 83.8%. On the other hand, the accuracy reported in the second row of Table 4 implies that  $Z_{location}$  are less informative features for other attributes. Similar results can also be observed for other attributes.

### 3.4 Disentangled Interpretability

We provide some examples to demonstrate that words having ambiguous or different aspects of semantics can be disentangled. Table 5 shows the results of nearby words. As can be seen, querying a word in  $Z_a$  with different attributes can help discover the ambiguous semantics implicitly encoded in the original word vectors  $X$ . The results also show that  $Z_{unseen}$  does capture meaningful information having little relevance to given attributes.

## 4 Application: Compact Features for Downstream Tasks

Here we demonstrate an application of the *modularity* and *compactness* properties of our DWE. We firstly aim to show the sub-embeddings can directly be informative features and can outperform GloVe with the same number of dimensions. With the high interpretability, selecting relevant

Query	Vectors	Nearby Words
turkey	$Z_{animal}$	geese, flock, goose
turkey	$Z_{location}$	greece, cyprus, ankara
mouse	$Z_{animal}$	mice, rat, rats
mouse	$Z_{artifact}$	keyboard, joystick, buttons
japan	$Z_{location}$	korea, vietnam, singapore
japan	$Z_{unseen}$	japanese, yakuza, yen
apple	$Z_{artifact}$	macintosh, software, mac
apple	$Z_{unseen}$	mango, cherry, tomato

Table 5: Results of nearby words.

sub-embeddings could be intuitive. Secondly, we will demonstrate that if deciding to fine-tune word vectors for a given downstream task, by using our DWE, we can focus on updating the relevant sub-embedding instead of the whole embedding. The advantage is that it reduces the number of learning parameters. Also, it could be regarded as a dimensional and interpretable regularization technique reducing overfitting.

We take a sentiment analysis task, IMDB movie review classification(Maas et al., 2011), for experiments. Intuitively,  $ADJ$  and  $ADV$  should be the most relevant attributes in  $\mathcal{A}$ . We then select 50 dimensions from  $Z \in \mathbb{R}^{300}$  with the lowest dropout rates in  $ADJ$  and  $ADV$  sub-areas for comparing with 50-dimensional GloVe<sup>4</sup> (GloVe-50). The embeddings with the selected dimensions are denoted by  $Z_{adj+adv}$ -50. When tuning our DWE with the classifier, we update the 52 dimensions ( $Z_{adj} \in \mathbb{R}^{23}$  and  $Z_{adv} \in \mathbb{R}^{29}$ ) of DWE and compare it with GloVe-300.

The document representations for classification is averaged word embeddings. The classifier is a logistic regression. When tuning the input word embeddings, we update the embeddings with gradient propagated from the classifier.

The results are listed in Table 6. From the table, we can see  $Z_{adj+adv}$ -50 directly outperforms GloVe-50 without tuning. A possible explanation is that GloVe-50 is forced to encode information less relevant to the sentiments, making it less effective than  $Z_{adj+adv}$ -50 in this task.

In the fine-tuning experiments, DWE can show slightly higher accuracy than GloVe-300 by updating only 52 instead of 300 dimensional features.

<sup>4</sup><https://nlp.stanford.edu/projects/glove/>

Feature	Without Tuning	After Tuning
GloVe-50	76.55	86.72
$Z_{adj+adv}$ -50	79.78	87.60
GloVe-300	<b>83.85</b>	87.72
DWE	83.67	<b>87.84</b>

Table 6: Classification accuracies (%) on IMDB dataset.

## 5 Conclusion

In this work, we propose a new definition and learning algorithm for obtaining disentangled word representations. As a result, the disentangled word vectors can show higher interpretability and preserve performance on various NLP tasks. We can also see the ambiguous semantics hidden in typical dense word embeddings can be extracted and separately encoded. Finally, we showed the disentangled word vectors can help generate compact and effective features for NLP applications. In the future, we would like to investigate whether similar effects can be found from non-distributional or contextualized word embeddings.

## References

- Ben Athiwaratkun and Andrew Gordon Wilson. 2017. [Multimodal word distributions](#). In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1645–1656.
- E. Bruni, N.-K. Tran, and M. Baroni. 2014. Multimodal distributional semantics. *J. Artif. Intell. Res.(JAIR)*, 49:1–47.
- Gino Brunner, Yuyi Wang, Roger Wattenhofer, and Michael Weigelt. 2018. [Natural language multitasking: Analyzing and improving syntactic saliency of hidden representations](#). *CoRR*, abs/1801.06024.
- José Camacho-Collados and Mohammad Taher Pilehvar. 2018. [From word to sense embeddings: A survey on vector representations of meaning](#). *J. Artif. Intell. Res.*, 63:743–788.
- A. Gladkova, A. Drozd, and S. Matsuoka. 2016. Analogy-based detection of morphological and semantic relations with word embeddings: what works and what doesn’t. In *Proceedings of the NAACL Student Research Workshop*, pages 8—15.
- Irina Higgins, David Amos, David Pfau, Sébastien Racanière, Loïc Matthey, Danilo J. Rezende, and Alexander Lerchner. 2018. [Towards a definition of disentangled representations](#). *CoRR*, abs/1812.02230.
- F. Hill, R. Reichart, and A. Korhonen. 2015. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(4):665–695.
- Sarthak Jain, Edward Banner, Jan-Willem van de Meent, Iain J Marshall, and Byron C. Wallace. 2018. [Learning disentangled representations of texts with application to biomedical abstracts](#). In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4683–4693. Association for Computational Linguistics.
- Durk P Kingma, Tim Salimans, and Max Welling. 2015. [Variational dropout and the local reparameterization trick](#). In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2575–2583. Curran Associates, Inc.
- Maria Larsson, Amanda Nilsson, and Mikael Kågebäck. 2017. [Disentangled representations for manipulation of sentiment in text](#). *CoRR*, abs/1712.10066.
- Omer Levy and Yoav Goldberg. 2014. [Neural word embedding as implicit matrix factorization](#). In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2177–2185. Curran Associates, Inc.
- Bofang Li, Tao Liu, Zhe Zhao, Buzhou Tang, Aleksandr Drozd, Anna Rogers, and Xiaoyong Du. 2017. Investigating different syntactic context types and context representations for learning word embeddings. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2421–2431.
- Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1*, pages 142–150. Association for Computational Linguistics.
- Igor Melnyk, Cáceres Nogueira dos Santos, Kahini Wadhawan, Inkit Padhi, and Abhishek Kumar. 2017. [Improved neural text attribute transfer with non-parallel data](#). *CoRR*, abs/1711.09395.
- Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. [Efficient estimation of word representations in vector space](#). *CoRR*, abs/1301.3781.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. [Distributed representations of words and phrases and their compositionality](#). In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

Dmitry Molchanov, Arsenii Ashukha, and Dmitry P. Vetrov. 2017. [Variational dropout sparsifies deep neural networks](#). In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 2498–2507.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *In EMNLP*.

Karl Ridgeway and Michael C. Mozer. 2018. [Learning deep disentangled embeddings with the f-statistic loss](#). In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 185–194. Curran Associates, Inc.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. [Introduction to the conll-2003 shared task: Language-independent named entity recognition](#). *CoRR*, cs.CL/0306050.

Yulia Tsvetkov, Manaal Faruqui, Wang Ling, Guillaume Lample, and Chris Dyer. 2015. Evaluation of word vector representations by subspace alignment. In *Proc. of EMNLP*.

# Multi-view Classification Model for Knowledge Graph Completion

Wenbin Jiang<sup>1</sup>, Mengfei Guo<sup>2\*</sup>,  
Yufeng Chen<sup>2</sup>, Ying Li<sup>1</sup>, Jinan Xu<sup>2</sup>, Yajuan Lyu<sup>1</sup>, Yong Zhu<sup>1</sup>

<sup>1</sup>Baidu Inc., Beijing, China

<sup>2</sup>School of Computer and Information Technology, Beijing Jiaotong University, Beijing, China

[{jiangwenbin, nicole, lvyajuan, zhuyong}@baidu.com](mailto:{jiangwenbin, nicole, lvyajuan, zhuyong}@baidu.com)

[{guomengfei, jaxu, chenyf}@bjtu.edu.cn](mailto:{guomengfei, jaxu, chenyf}@bjtu.edu.cn)

## Abstract

Most previous work on knowledge graph completion conducted single-view prediction or calculation for candidate triple evaluation, based only on the content information of the candidate triples. This paper describes a novel multi-view classification model for knowledge graph completion, where multiple classification views are performed based on both content and context information for candidate triple evaluation. Each classification view evaluates the validity of a candidate triple from a specific viewpoint, based on the content information inside the candidate triple and the context information nearby the triple. These classification views are implemented by a unified neural network and the classification predictions are weightedly integrated to obtain the final evaluation. Experiments show that, the multi-view model brings very significant improvements over previous methods, and achieves the new state-of-the-art on two representative datasets. We believe that, the flexibility and the scalability of the multi-view classification model facilitates the introduction of additional information and resources for better performance.

## 1 Introduction

Knowledge graph (KG) is a typical kind of graph-structured knowledge base (KB). Nowdays, there exist many famous KGs such as YAGO (Suchanek et al., 2007), Freebase (Bollacker et al., 2008) and DBpedia (Lehmann et al., 2015). Large-scale KGs are widely used in many applications such as semantic searching (Kasneci et al., 2008; Schuhmacher and Ponzetto, 2014; Xiong et al., 2017), question answering (Zhang et al., 2016; Hao et al., 2017) and machine reading (Yang and Mitchell,

2017). A KG contains a set of triples indicating facts, each of which is composed of a *head* entity, a *tail* entity, and a *relation* indicating the relationship between the two entities. It is nearly impossible to collect a complete set of facts or triples for a KG, especially in open domains. In fact, many valuable valid triples are missing even for the existing well-built large-scale KGs such as Freebase (Socher et al., 2013; West et al., 2014). Many researchers devote their efforts to the problem of knowledge graph completion (KGC), the core operation of which is to evaluate the validity of candidate triples.

Previous work on KGC mainly include two groups, embedding-based methods and classification-based methods. Embedding-based models learn embeddings for entities and relations, and evaluate candidate triples based on the embeddings and specific distance metrics. Representative models include TransE (Bordes et al., 2013) and its extensions (Wang et al., 2014; Lin et al., 2015b; Ji et al., 2015; Nguyen et al., 2016), DistMult (Yang et al., 2015) and ComplEx (Trouillon et al., 2016). Classification-based models learn neural networks to evaluate the validity of candidate triples. Representative models include ConvE (Dettmers et al., 2018) and ConvKB (Nguyen, 2017). The major advantage of classification-based methods is that they directly model the evaluation of the validity of candidate triples, probably leading to better performance. Most of these previous work conducted single-view prediction based on content information, that is, evaluating a candidate triple according to a single distance metric or classification schema, resorting to information restricted in the scope of the candidate triple. We believe that multiple learning views for triple evaluation as well as context information of the candidate triple would contribute to better performance.

In this work, we propose for KGC a novel multi-view classification model, where multiple classifi-

\*Joint first author. Guo participated in the optimization of this work during the internship in Baidu.

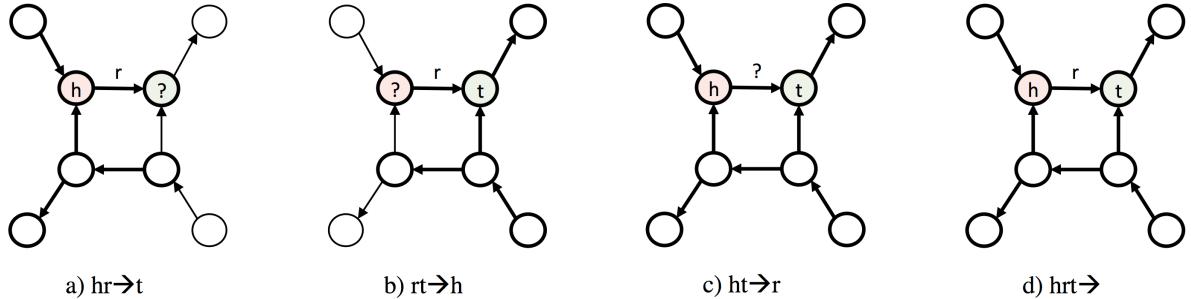


Figure 1: Illustration of sub-graphs corresponding to the learning views. The colored nodes indicate the head and tail entities of the candidate triple. The bold nodes and edges are the elements in the retrieved sub-graphs. The question marks indicate the elements to be predicted.

cation views are performed to estimate the validity of a candidate triple, based on both content and context information of the triple. There are four classification views for candidate triple evaluation. Each of the first three views performs component prediction, where a specific component of the candidate triple is predicted according to the other two components as well as its nearby triples. The last view performs plausibility prediction, where the plausibility of the candidate triple is predicted according to its components as well as its nearby triples. The prediction conditions of these views investigate both content and context information of the candidate triple, that is, the components in the candidate triple, and the triples nearby the candidate triple. The content and context information can be represented as a sub-graph surrounding the candidate triple. These classification views are implemented by a unified neural network with shared embedding and encoding layers and separated prediction layers, and the classification predictions are integrated by a weighted integration procedure for better candidate triple evaluation. In the unified neural network, the sub-graphs indicating the content and context of the candidate triples are encoded in a sequential manner, by converting the sub-graphs into sequential tree representations. It facilitates the utilization of advanced encoders such as BiLSTM or Transformer.

We experiment on two widely used benchmark datasets, FB15k-237 and WN18RR, specific versions of Freebase and WordNet. We find that the multi-view model achieves the new state-of-the-art, significantly outperforming previous work on KGC. We also find that we can promote the efficiency of the multi-view model in realistic applications, by a coarse-to-fine strategy where the first two views are performed to give a list of candidates, and the

overall model is then performed to evaluate these candidates. We believe that, the flexibility and the scalability of the multi-view classification model facilitates the introduction of additional information and resources for better performance.

## 2 Related Work

Most existing KGC models are based on KG embeddings, which aims at learning distributed representations for entities and relations in a KG. In these models, the candidate triples are evaluated by some specific distance metrics based on the embeddings. These models perform embedding learning with local information in individual triples, including translation-based models (Bordes et al., 2013; Wang et al., 2014; Lin et al., 2015b), semantic matching models (Yang et al., 2015; Nickel et al., 2016; Trouillon et al., 2016), and neural network models (Dettmers et al., 2018; Jiang et al., 2019; Nguyen, 2017). There also exist KGC models based on classification, where classifiers are learnt to evaluate the validity of candidate triples (Dettmers et al., 2018; Nguyen, 2017). Both kinds of previous work consider only one view, with simple distance metrics and classification operations. In contrast, multi-view learning enables the incorporation of much more views that utilize internal and external information for triple evaluation.

In recent years, many efforts were devoted to embedding learning based on non-local information such as multi-hop paths (Lin et al., 2015a; Das et al., 2017) and  $k$ -degree neighborhoods (Feng et al., 2016; Schlichtkrull et al., 2017). Some researchers also investigated graph embeddings in social network and other areas (Perozzi et al., 2014; Grover and Leskovec, 2016; Ristoski and Paulheim, 2016; Cochez et al., 2017). Compared with these work, our method not only learns em-

View Type	Instance from $g_v$	Instance from $g_v^-$
<b>hr→t</b>	$g_{hr \rightarrow t} = \langle \mathcal{G}(h, r, ?), t \rangle$	$g_{hr \rightarrow t}^- = \langle \mathcal{G}(\mathcal{S}(h, r, ?)), \text{none} \rangle, \text{ s.t. } \mathcal{S}(h, r, ?) \notin \mathcal{KG}$
<b>rt→h</b>	$g_{rt \rightarrow h} = \langle \mathcal{G}(?, r, t), h \rangle$	$g_{rt \rightarrow h}^- = \langle \mathcal{G}(\mathcal{S}(?, r, t)), \text{none} \rangle, \text{ s.t. } \mathcal{S}(?, r, t) \notin \mathcal{KG}$
<b>ht→r</b>	$g_{ht \rightarrow r} = \langle \mathcal{G}(h, ?, t), r \rangle$	$g_{ht \rightarrow r}^- = \langle \mathcal{G}(\mathcal{S}(h, ?, t)), \text{none} \rangle, \text{ s.t. } \mathcal{S}(h, ?, t) \notin \mathcal{KG}$
<b>hrt→</b>	$g_{hrt \rightarrow} = \langle \mathcal{G}(h, r, t), \text{true} \rangle$	$g_{hrt \rightarrow}^- = \langle \mathcal{G}(\mathcal{S}(h, r, t)), \text{false} \rangle, \text{ s.t. } \mathcal{S}(h, r, t) \notin \mathcal{KG}$

Table 1: Instance generation for each learning view. The first/second part in an instance is used as the input/output for classification. The function  $\mathcal{G}$  retrieves the sub-graph surrounding the candidate triple with the maximum height and width limitations. The function  $\mathcal{S}$  receives a tuple and returns a randomly corrupted tuple that not exists in the KG, by randomly replacing a known component which is not denoted by the question mark. The operator  $\in$  indicates that a tuple is *equal to* or *inside of* a triple.

beddings for individual entities and relations based on non-local information, but also obtains representations for sub-graphs resorting to complicated neural encoders. This manner probably brings better KGC performance by leveraging global information more effectively.

### 3 Method: Multi-view Classification

A knowledge graph  $\mathcal{KG}$  contains a set of triples indicating facts,  $\{(h, r, t)\} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ . Each triple  $(h, r, t)$  consists of two entities  $h$  and  $t$  referred to the subject and object of the triple, and a relation  $r$  referred to the relationship between the two entities.  $\mathcal{E}$  and  $\mathcal{R}$  indicates the possible entity set and the possible relation set, respectively. The fundamental problem for KGC is to define a candidate triple evaluation model  $f : \mathcal{E} \times \mathcal{R} \times \mathcal{E} \rightarrow \mathbb{R}$ , giving each candidate triple  $(h, r, t)$  a score indicating the validity of the triple.

#### 3.1 Classification Views

We adopt a multi-view classification model for KGC, where a candidate triple is evaluated from four different views. The first three views adopt the generative methodology, each view predicts a specific component of the candidate triple according to the other two components and the nearby triples. The last view adopts the discriminative methodology, it predicts the plausibility of the whole triple according to its components as well as its nearby triples. In the prediction conditions of these views, the components in the candidate triple are content information inside the triple, and the triples nearby the candidate triple are context information outside the triple.

In details, the first view **hr→t** predicts  $t$  based on  $h, r$  and their context, the second view **rt→h** predicts  $h$  based on  $r, t$  and their context, the third view **ht→r** predicts  $r$  based on  $h, t$  and their context, and the fourth view **hrt→** predicts the plausibility given  $h, r, t$  and their context. We denote the view set as  $\mathcal{V}$ , containing the four views mentioned above. These views evaluate the candidate triple from different viewpoints and can be integrated to give better prediction.

In the prediction condition of each view, the context information includes the entities and relations nearby the candidate triple, and excludes the entities and relations that can only be reached by way of the entity or relation to be predicted. The content and context can be jointly represented as the sub-graph surrounding the candidate triple. For each of the first three views, the entity or relation to be predicted is replaced by a specific placeholder. The sub-graph can be extracted by breadth-first traversal from the candidate triple, without passing by the entity or relation to be predicted. In the traversal procedure, two hyperparameters  $d$  and  $w$  are introduced to restrict the depth and width of the sub-graph. Specifically,  $d$  defines the maximum distance between an entity and the candidate triple, and  $w$  defines the maximum branch count when passing by an entity.

The sub-graphs can be linearized as sequences of symbols with paired brackets in specific positions. The linearization facilitates the sequential encoding of graphic structures, which is proved to be effective and efficient in syntactic parsing. Table 1 shows the learning views and Figure 1 shows the content and context information for each view.

#### 3.2 Instance Generation

Given a learning view  $v \in \mathcal{V}$ , we define a pair of instance generation functions,  $g_v$  and  $g_v^-$ , to generate positive and negative classification instances for a candidate triple under this view. The instances are used as classification instances for triple evaluation. In an instance  $\langle x, y \rangle$ , the source part  $x$  is a linearized sequence representing a sub-graph, and the target part  $y$  is a label indicating an entity, a

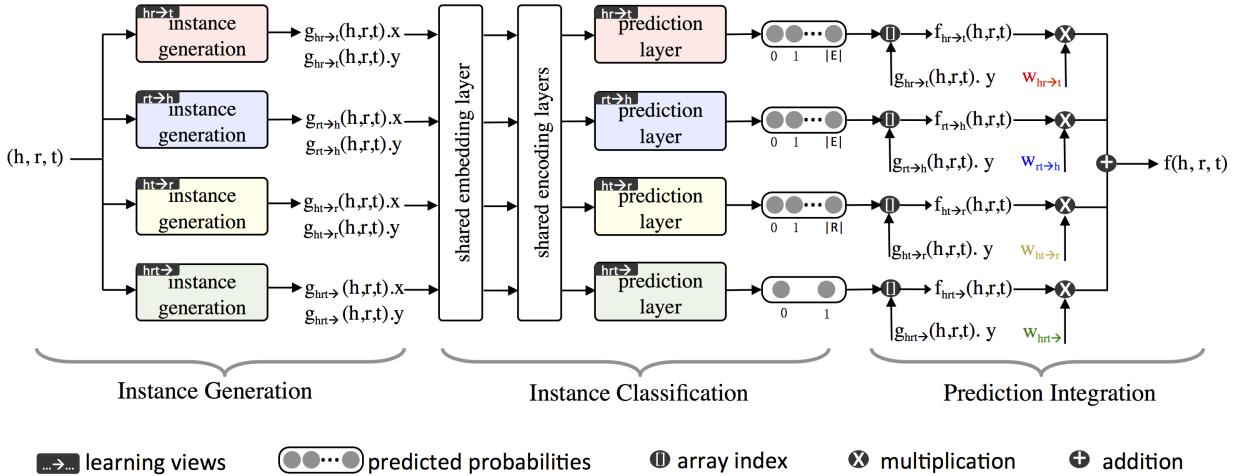


Figure 2: The overall multi-task learning architecture for the multi-view learning model.

relation or a boolean symbol. They correspond to the input and output for the learning of the classification models. For a given triple and a given view  $v$ , we always generate one positive view instance, but only generate a negative instance with a certain frequency  $\rho_v$ . The frequencies for the first three views should be much smaller than 1 in order to balance the instances with respect to the classification labels.

The positive instances are generated directly according to the schemas of the views. The negative instances are necessary for the learning of the triple evaluation model especially for the forth view. The source part of a negative instance can be generated by replacing a random component in the tuple with a random symbol of the same type, to satisfy the condition that the changed tuple is not *equal to* or *inside of* a triple in the KG. The target part for a negative instance is **none** for the first three views, and **false** for the fourth view. Table 1 shows the instance generation functions and their instances.

For each learning view, both positive and negative instances generated from the training triples are used for training, while only positive instances generated from the candidate triple are needed for testing. The classification models for the learning views can be trained with separated classifiers or in a multi-task framework. To promote the information sharing and interaction between learning views, we realized the multi-view model in a multi-task learning architecture, where each sub-task takes charge of a specific learning view. In the multi-task architecture, the instances for a training

or testing triple are simultaneously assigned to the sub-tasks according to their corresponding views. The details for realization will be described in the next section.

### 3.3 Triple Evaluation

Given a candidate triple, four classification instances are generated for the learning views by the corresponding positive instance generation functions. The evaluation given by each learning view is obtained by evaluating the corresponding instance with the corresponding classifier. The evaluation given by the whole multi-view model is the weightedly summation of the evaluations given by these views:

$$f(h, r, t) = \sum_{v \in \mathcal{V}} \mathbf{w}_v f_v(h, r, t)$$

The function  $f_v$  and the hyperparameter  $\mathbf{w}_v$  indicate the view-specific evaluation function and its weighting coefficient, respectively.

The view-specific evaluation function invokes the classification model of the view with the source part of the instance as input, and returns the prediction score corresponding to the target part of the instance:

$$f_v(h, r, t) = \sum_{v \in \mathcal{V}} \mathbf{w}_v \mathcal{F}_v(g_v^+(h, r, t) \cdot x) [g_v^+(h, r, t) \cdot y]$$

The function  $\mathcal{F}$  indicates the classification procedure of the sub-task corresponding to a specific learning view, it takes the source part of the instance as input and gives the prediction scores

on all possible labels. The operator  $\cdot$  indexes the source or target part of the instance, and the operator  $[ ]$  indexes the score corresponding to the target part.

For each triple in the testing set, we should compare its validity with those of the candidate triples, which are generated by replacing the head or tail entity with another entity. This means that, for a KG with millions of entities, millions of candidate triples should be evaluated by the multi-view model for each testing triple. To promote the efficiency of the multi-view model, we adopt a coarse-to-fine strategy in testing, where the first or second view is performed to give a list of  $k$ -best candidates, and the overall model is then performed to evaluate these candidates.

## 4 Realization: Multi-task Architecture

We implement the multi-view learning in a multi-task architecture, where each sub-task takes charge of a specific learning view. The multi-task learning strategy enables information sharing and interaction between the sub-tasks, thus leading to better performance.

### 4.1 Overall Pipeline

We design a unified neural multi-task learning architecture for the multi-view model. The overall procedure of the multi-task architecture is shown in Figure 2. The overall procedure is composed of three stages, instance generation, instance classification and prediction integration. The instance generation stage takes as input the given triple, and generates classification instances for all learning views by the instance generation functions. The instance classification stage takes as input the source parts of these instances, and predicts the labels for each input with the corresponding view-specific classification model. The prediction integration stage takes as input the predictions of all the classification models, and computes the overall training cost and evaluation score according to the target parts of the instances. Note that we need not compute the overall evaluation score for training, nor generate the negative instances for testing.

In the instance classification stage, all the classification models follow the same pipeline composed of embedding, encoding and predicting. For predicting, these models adopt separated predicting layers due to their essentially different learning objects. For embedding and encoding, these models

adopt the shared layers following the conventional strategy in NLP multi-task learning work. This is reasonable because the relationship between an instance and its components is analogous to that between a sentence and its words. The architecture in Figure 2 shows the multi-task learning architecture with shared embedding and encoding layers.

We add a specific symbol indicating the learning view at the beginning of the source part of the instance. This is similar to the idea in multilingual NMT that a specific markup is added at the beginning of a source language sentence to indicate the target language. The marked source parts of the instances are input into the same encoding layer. According to the added markups, the neural network learns and applies different information propagation regularities for instances of different views, while sharing network parameters as much as possible.

### 4.2 Neural Classifier

We use multi-layer Transformer as the encoding layers and logistic regression with softmax as the classification layers. Given the source part of an instance,  $x = (x_1, x_2, \dots, x_n)$ , which is a sequence of entities and relations with paired brackets indicating an linearized sub-graph, we construct the representation for each element  $x_i \in x$  as:

$$\mathbf{h}_i^0 = \mathbf{x}_i^e + \mathbf{x}_i^p$$

where  $x_i^e$  is the element embedding and  $x_i^p$  the position embedding, indicating the current element and its position in the sequence, respectively. We feed these representations into a stack of  $L$  successive Transformer encoders as:

$$\mathbf{h}_i^l = \text{Transformer}(\mathbf{h}_i^{l-1}), l = 1, 2, \dots, L$$

where  $\mathbf{h}_i^l$  is the hidden state of  $x_i$  after the  $l$ -th encoding layer. We omit the detailed description of Transformer since it is already ubiquitous recently.

The representation used for the subsequent classification layer is the concatenation of the final hidden states corresponding to the components of the triple for evaluation. Note that for the first three views, one of the three components is a placeholder. The training procedure aims to find the parameters minimizing the cross-entropy loss:

$$\mathcal{L}(\theta) = \sum_{z \in \mathcal{KG}} \sum_{v \in \mathcal{V}} \sum_{\langle x, y \rangle \in \{g_v^+(z), g_v^-(z)\}} \mathcal{C}(\mathcal{F}_v(x, \theta), y)$$

Setting	FB15k-237						WN18RR					
	Content			+Context			Content			+Context		
	MR	MRR	H@10	MR	MRR	H@10	MR	MRR	H@10	MR	MRR	H@10
$\mathcal{V}$ - hr→t	161	.267	.431	209	.289	.485	2420	.408	.477	2262	.412	.498
$\mathcal{V}$ - rt→h	155	.277	.443	178	.296	.476	3318	.377	.437	3573	.393	.473
$\mathcal{V}$ - ht→r	150	.294	.468	215	.310	.481	2824	.424	.491	2713	.462	.522
$\mathcal{V}$ - hrt→	156	.290	.475	161	.335	.492	3011	.421	.477	2713	.436	.509
$\mathcal{V}$	139	.330	.491	151	.359	.521	2193	.446	.526	2210	.484	.540

Table 2: The contributions of the individual views to the overall model, evaluated on the development sets.

		FB15k-237	WN18RR
Statistics	# entity	14,541	40,943
	# relation	237	11
Partition	Train	272,115	86,835
	Develop	17,535	3,034
	Test	20,466	3,134

Table 3: The statistics of FB15k-237 and WN18RR, including number of entities, relations, and triples in each partition.

Here, we use  $\mathcal{F}$  to indicate the feedforward procedure,  $\mathcal{C}$  to indicate the cross-entropy cost function, and  $\mathcal{KG}$  to indicate the set of training triples. In the testing procedure, only positive instances are used for a testing triple. The testing procedure evaluates a triple by integrating the four views as mentioned before.

## 5 Experiments

### 5.1 Datasets and Evaluation Protocol

We evaluate the multi-view model on two widely used benchmark datasets, FB15k-237 and WN18RR, which are subsets of two common datasets FB15k and WN18. The original FB15k and WN18 are easy for KGC due to the reversible relations, it could not reflect the real performance of KGC models. Therefore, researchers create FB15k-237 and WN18RR to fix the reversible relation problem, and make the KGC task more realistic (Toutanova and Chen, 2015; Dettmers et al., 2018). The statistics of the datasets are summarized in Table 3.

The purpose of KGC is to predict a missing entity given a relation and another entity. Following Bordes et al. (2013), for every testing triple, we replace the head or tail entities with all entities existed in the knowledge graph, and rank these triples in ascending order according to the triple evaluation function, following the *filtered* setting protocol

which does not consider any corrupted triples that appear in the original KG. Following (Nguyen, 2017), we use three common evaluation metrics, mean rank (MR), mean reciprocal rank (MRR), and the proportion of the valid test triples ranking in top  $n$  predictions (H@n) with  $n \in \{1, 3, 10\}$ .

### 5.2 Details for Training and Testing

The multi-view model is trained with instances generated from the training triples, and is used to evaluate the instances generated from the testing triples. There are parameters to be tuned in the procedures of instance generation, model training, and model testing.

For the definition of the subgraph indicating the content and context of a triple, the maximum depth  $d$  and width  $w$  will be determined in the developing procedure. For the transformer used for classification, the number of Transformer blocks is  $L = 6$ , the number of self-attention heads is  $A = 4$ , and the hidden size and the feed-forward size are  $D = 256$  and  $2D = 512$ , respectively. The dropout strategy is applied on embedding and encoding layers with dropout rate 0.5. We adopt the Adam algorithm (Kingma and Ba, 2014) for tuning with a learning rate  $\eta = 5 \times 10^{-4}$ . The multi-view model is trained with batch size  $B = 256$  for at most 1000 epochs. For the coarse-to-fine prediction strategy in the testing procedure, the number  $k$  of best candidates given by the first or second view is determined on the development set. We choose  $\rho = [0.001, 0.001, 0.01, 1.0]$  for negative instance generation,  $d = 2$  and  $w = 3$  for sub-graph retrieval, and  $w = [0.30, 0.30, 0.25, 0.15]$  for view combination by grid search experiments on development sets. The above models are implemented on PaddlePaddle<sup>1</sup>.

<sup>1</sup> <https://github.com/PaddlePaddle/Paddle>

Model	FB15k-237					WN18RR				
	MR	MRR	H@1	H@3	H@10	MR	MRR	H@1	H@3	H@10
R-GCN+	-	.249	.151	.264	.417	-	-	-	-	-
KB-LRN	209	.309	.219	-	.493	-	-	-	-	-
ConvE	246	.316	.239	.350	.491	5277	.460	.390	.430	.480
ConvR	-	.350	.261	.385	.528	-	.475	.443	.489	.537
RotatE	177	.338	.241	.375	.533	3340	.476	.428	.492	.571
TuckER	-	.358	.266	.394	.544	-	.470	.443	.482	.526
pLogicNet	173	.332	.237	.367	.524	3408	.441	.398	.446	.537
SimpleClassification	161	.307	.223	.382	.525	2193	.446	.393	.456	.522
MultiView	<b>134</b>	.320	<b>.276</b>	<b>.412</b>	<b>.544</b>	<b>1738</b>	.463	<b>.462</b>	<b>.494</b>	.549

Table 4: Performance of multi-view learning compared with previous methods, on the testing sets of FB15k-237 and WN18RR. R-GCN+: (Schlichtkrull et al., 2017), KB-LRN: (Garcia-Duran and Niepert, 2017), ConvE: (Dettmers et al., 2018), ConvR: (Jiang et al., 2019), RotatE: (Sun et al., 2019), TuckER: (Balažević et al., 2019), pLogicNet: (Qu and Tang, 2019). SimpleClassification: multi-view model based on simple classification ( $\mathbf{hr} \rightarrow \mathbf{t}$  and  $\mathbf{rt} \rightarrow \mathbf{h}$ ), MultiView: multi-view model with all components ( $\mathbf{hr} \rightarrow \mathbf{t} + \mathbf{rt} \rightarrow \mathbf{h} + \mathbf{ht} \rightarrow \mathbf{r} + \mathbf{hrt} \rightarrow$ ) .

### 5.3 Main Results and Analysis

We verify the effectiveness of the multi-view model, by investigating the contributions of the learning views to the overall model. Table 2 shows the performance on the development sets of the two datasets. Note that for each experimental setting, the model is retrained on the classification instances generated according to the views in the setting. We find that each of the learning views contributes to the final performance, and context information brings further improvement.

The performance of the multi-view model on the testing sets of the two datasets is shown in Table 4, where the performance of methods in previous work is also listed. The multi-view learning model achieves the new state-of-the-art on both benchmark datasets. Compared with previous work, it gives significantly better MR on both datasets. It reveals that in the multi-view model, the answers are high in the ranked lists on average. Considering that it does not use any optimization tricks, we think that it still has potential for further improvement by introducing additional information and resources, such as pre-trained embeddings, text descriptions and surface morphologies of entities and relations. We also find that, the simple classification model based on the first two views, which brutally predict the head and tail entities according to the rest components, achieves very promising results. In other words, the first two views lead to simple but effective classification-based KGC models.

The simple classification model works very fast in evaluation of candidate triples, since direct pre-

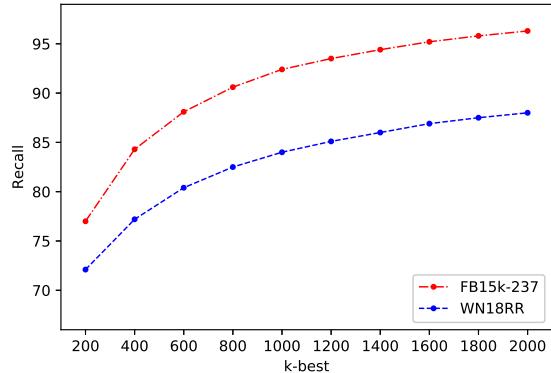


Figure 3: The recall curves of  $k$ -best pre-filtering.

diction of the missing entities is equivalent to evaluating thousands of candidate triples simultaneously. We can adopt a coarse-to-fine strategy in realistic applications. It pre-selects the  $k$ -best candidates by the first two views, and reranks the candidates by the whole multi-view model. Figure 3 shows the experimental results. The quality of the candidate list is measured with recall, indicating the percentage of the instances for which the candidate lists contain the answers. We find that the pre-selection of 2000-best list achieves very high recalls on the two datasets, especially on FB15k-237. Therefore, we can safely filter out most of the candidates with little loss of final precision. It facilitates the introduction of more features in the multi-view model by restricting the search space to a small but precise  $k$ -best list.

## 6 Conclusion

We propose a novel multi-view classification model for knowledge graph completion, where multiple classification views are performed based on both content and context information for candidate triple evaluation. The multi-view model is implemented with a simple and unified multi-task learning architecture where the parameters are shared across all the learning views. It achieves the new state-of-the-art although without using any optimization tricks. The multi-view model can be improved from two perspectives in the future. First, the multi-view model can leverage more kinds of information and resources for better performance, such as the descriptions of the entities and relations, as well as related information in external knowledge bases. Second, the multi-task learning architecture can introduce different kinds of neural networks to better model different kinds of information, for example, sequential neural networks for sequences and graph neural networks for graphs.

## Acknowledgments

The authors Chen and Xu were also supported by the National Nature Science Foundation of China (No. 61876198, 61976015, 61370130 and 61473294), the Fundamental Research Funds for the Central Universities (No. 2018YJS025), the Beijing Municipal Natural Science Foundation (No. 4172047), and the International Science and Technology Cooperation Program of China under Grant No. K11F100010. We sincerely thank Quan Wang in Baidu for the enlightening suggestions in the research procedure, and the anonymous reviewers for their valuable comments and suggestions.

## References

- Ivana Balažević, Carl Allen, and Timothy M. Hospedales. 2019. TuckER: Tensor factorization for knowledge graph completion. In *Proceedings of EMNLP-IJCNLP*.
- Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of ICMD*.
- Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Proceedings of NIPS*.
- Michael Cochez, Petar Ristoski, Simone Paolo Ponzetto, and Heiko Paulheim. 2017. Global rdf vector space embeddings. In *Proceedings of ISWC*.
- Rajarshi Das, Arvind Neelakantan, David Belanger, and Andrew McCallum. 2017. Chains of reasoning over entities, relations, and text using recurrent neural networks. In *Proceedings of EACL*.
- Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Proceedings of AAAI*.
- Jun Feng, Minlie Huang, Yang Yang, and Xiaoyan Zhu. 2016. Gake: Graph aware knowledge embedding. In *Proceedings of COLING*.
- Alberto Garcia-Duran and Mathias Niepert. 2017. Kblrn: End-to-end learning of knowledge base representations with latent, relational, and numerical features. In *arXiv preprint abs/1709.04676*.
- Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of SIGKDD*.
- Yanchao Hao, Yuanzhe Zhang, Kang Liu, Shizhu He, Zhanyi Liu Hua Wu, and Jun Zhao. 2017. An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge. In *Proceedings of ACL*.
- Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of ACL-IJCNLP*.
- Xiaotian Jiang, Quan Wang, and Bin Wang. 2019. Adaptive convolution for multi-relational learning. In *Proceedings of NAACL-HLT*.
- Gjergji Kasneci, Fabian M Suchanek, Georgiana Ifrim, Maya Ramanath, and Gerhard Weikum. 2008. Naga: Searching and ranking knowledge. *Proceedings of ICDE*.
- Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. In *arXiv preprint arXiv:1412.6980*.
- Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Soren Auer, and et al. 2015. Dbpedia-a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*.
- Yankai Lin, Zhiyuan Liu, Huanbo Luan, Maosong Sun, Siwei Rao, and Song Liu. 2015a. Modeling relation paths for representation learning of knowledge bases. In *Proceedings of EMNLP*.
- Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015b. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of AAAI*.

- Dat Quoc Nguyen. 2017. An overview of embedding models of entities and relationships for knowledge base completion. In *arXiv:1703.08098*.
- Dat Quoc Nguyen, Kairit Sirts, Lizhen Qu, and Mark Johnson. 2016. Stranse: a novel embedding model of entities and relationships in knowledge bases. In *Proceedings of NAACL-HLT*, pages 460–466.
- Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. 2016. Holographic embeddings of knowledge graphs. In *Proceedings of AAAI*.
- Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of SIGKDD*.
- Meng Qu and Jian Tang. 2019. Probabilistic logic neural networks for reasoning. *arXiv:1906.08495*.
- Petar Ristoski and Heiko Paulheim. 2016. Rdf2vec: Rdf graph embeddings for data mining. In *Proceedings of ISWC*.
- Michael Schlichtkrull, Thomas Kipf, Peter Bloem, Ivan Titov Rianne van den Berg, and Max Welling. 2017. Modeling relational data with graph convolutional networks. In *arXiv:1703.06103*.
- Michael Schuhmacher and Simone Paolo Ponzetto. 2014. Knowledge-based graph document modeling. In *Proceedings of WSDM*, pages 543–552.
- Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. *NIPS*, pages 926–934.
- Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: A core of semantic knowledge. *Proceedings of WWW*, pages 697–706.
- Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. *arXiv:1902.10197*.
- Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *Proceedings of the ACL Workshop on Continuous Vector Space Models and their Compositionality*, pages 57–66.
- Theo Trouillon, Johannes Welbl, Sebastian Riedel, Eric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. *Proceedings of ICML*, pages 2071–2080.
- Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of AAAI*, pages 1112–1119.
- Robert West, Evgeniy Gabrilovich, Kevin Murphy, Shaohua Sun, Rahul Gupta, and Dekang Lin. 2014. Knowledge base completion via search-based question answering. In *Proceedings of WWW*, pages 515–526.
- Chenyan Xiong, Russell Power, and Jamie Callan. 2017. Explicit semantic ranking for academic search via knowledge graph embedding. *Proceedings of WWW*, pages 1271–1279.
- Bishan Yang and Tom Mitchell. 2017. Leveraging knowledge bases in lstms for improving machine reading. In *Proceedings of ACL*.
- Bishan Yang, Wen tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *Proceedings of ICLR*.
- Yuanzhe Zhang, Kang Liu, Shizhu He, Guoliang Ji, Zhanyi Liu, Hua Wu, and Jun Zhao. 2016. Question answering over knowledge base with neural attention combining global knowledge information. *arXiv:1606.00979*.

# Knowledge-Enhanced Named Entity Disambiguation for Short Text

Zhifan Feng, Qi Wang, Wenbin Jiang, Yajuan Lyu, Yong Zhu

Baidu Inc., Beijing, China

{fengzhifan, wangqi31, jiangwenbin, lvyajuan, zhuyong}@baidu.com

## Abstract

Named entity disambiguation is an important task that plays the role of bridge between text and knowledge. However, the performance of existing methods drops dramatically for short text, which is widely used in actual application scenarios, such as information retrieval and question answering. In this work, we propose a novel knowledge-enhanced method for named entity disambiguation. Considering the problem of information ambiguity and incompleteness for short text, two kinds of knowledge, factual knowledge graph and conceptual knowledge graph, are introduced to provide additional knowledge for the semantic matching between candidate entity and mention context. Our proposed method achieves significant improvement over previous methods on a large manually annotated short-text dataset, and also achieves the state-of-the-art on three standard datasets. The short-text dataset and the proposed model will be publicly available for research use.

## 1 Introduction

Name entity disambiguation (NED) aims to associate each entity mention in the text with its corresponding entity in the knowledge graph (KG). It plays an important role in many text-related artificial intelligent tasks such as recommendation and conversation, since it works as a bridge between text and knowledge. In decades, researchers devoted their efforts to NED in many ways, including the rule-based methods (Shen et al., 2014), the conventional statistic methods (Shen et al., 2014) and the deep learning methods (Octavian-Eugen Ganea, 2017). On formal text, state-of-the-art methods achieve high performance thanks to the well-written utterance and rich context. However, experiments show that the performance of these methods degrades dramatically on informal text,

for example, the short text widely used in many real application scenarios such as information retrieval and human-machine interaction. It is difficult for existing methods to make decisions on the non-standard utterance without adequate context.

The discrimination procedure of NED depends on sufficient context in the input text, which is usually noisy and scarce in the short text used in information retrieval and human-machine interaction. For example, an analysis based on search engine logs demonstrates that a search query contains 2.35 words on average (Yi Fang, 2011). Such short text could not provide adequate context which is necessary for NED models. In recent years, many efforts improve NED by exploiting more powerful models and richer context information (Shen et al., 2014). These methods mainly focus on the better utilization of existing context. Therefore, they can not improve NED effectively on short text since the problem of information shortage still exists. Intuitively, it is hard for NED to achieve essential improvement on short text if it can not exploit external information to enhance the recognition procedure.

In information scarce situations, human beings can still perform recognition by association with related external information, such as commonsense or domain-specific knowledge. It inspires us that, NED on short text could be improved if appropriate external knowledge can be retrieved and considered. We propose a novel knowledge-enhanced NED model, where the prediction procedure of NED is enhanced by two kinds of knowledge formalized as two different KGs. The one kind is conceptual knowledge formalized as a conceptual KG, it is used to augment the representation of the entity mention by giving the mention a concept embedding. The other kind is factual knowledge formalized as a factual KG, it is used to augment the representation of each candidate entity by giving the entity an entity embedding. The augmented

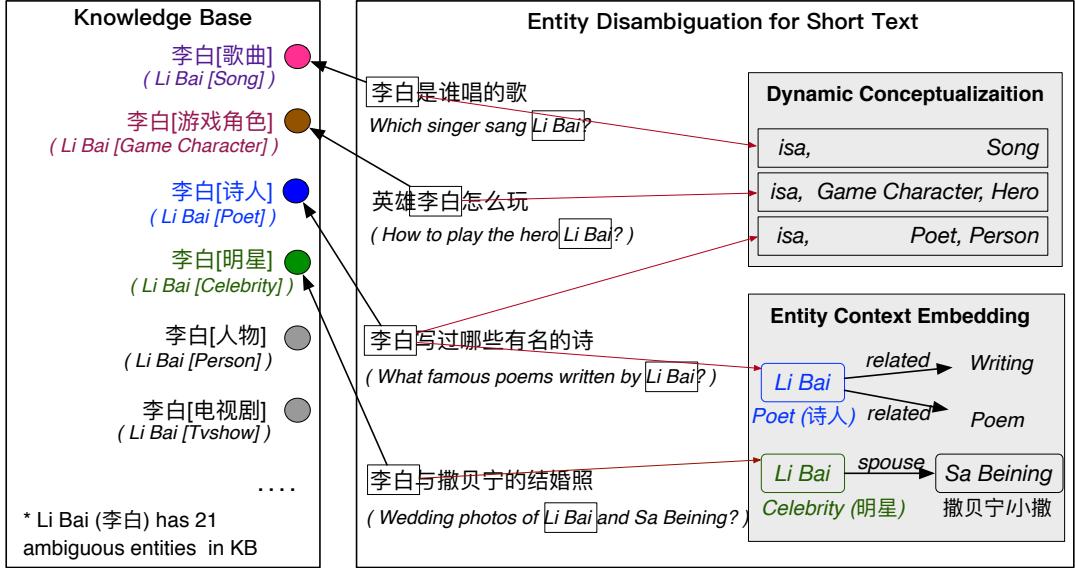


Figure 1: Short text entity disambiguation and our method. We solve the problem of entity disambiguation of sparse short texts through dynamic conceptualization and entity context embedding.

representations of the mention context and the candidate entities are used in a matching network for better NED prediction.

We validate the knowledge-enhanced NED model on three public NED datasets for short text (NEEL, KORE50 and FUDAN) as well as our new dataset (**DUEL**), which is constructed for information acquiring scenarios and will be publicly available for research use. Experiments show that the knowledge-enhance NED model performs significantly better than previous methods. It shows the effectiveness of external knowledge in improving the prediction of NED in information scarce situations. The contribution of our work includes two aspects. First, we introduce conceptual and factual knowledge to improve NED for short text for the first time, and achieve significant improvement. Second, we release a large-scale good-quality NED dataset for short text for information acquisition scenarios, which is complementary existing datasets.

The rest of this paper is organized as follows. We first introduce the NED task and the baseline method (section 2), and then describe the architecture and details of our knowledge-enhanced model (section 3). After giving the detailed experimental analysis (section 4), we give the related work (section 5) and conclude the work.

## 2 Task Definition and Baseline Model

NED is a fundamental task in the area of natural language processing and knowledge base. It aims to

associate each entity mention in the given text with its corresponding entity in the given KG. Formally, given a KG  $\mathcal{G}$  and a piece of text  $\mathcal{T}$ , it assigns each mention  $m \in \mathcal{T}$  with an entity  $e \in \mathcal{G}$  indicating that  $m$  refers to  $e$ , or with the symbol  $\phi$  indicating that there is no corresponding entity.

The disambiguation procedure can be formalized as matching between the context of the mention and each candidate entities.

$$\mathbf{f}(m) = \begin{cases} \arg \max_{e \in \mathbf{e}(m)} (\mathbf{s}(e, \mathbf{c}(m))), & \mathbf{e}(m) \neq \emptyset \\ \phi, & \text{otherwise} \end{cases} \quad (1)$$

Here, the function  $\mathbf{e}$  returns the entity candidate set for a given mention, and the function  $\mathbf{c}$  returns the context of the given mention. The function  $\mathbf{s}$  is used to evaluate the matching degree between context and candidate, and is usually implemented as matching networks. If the entity candidate set is empty or the highest matching score is below a given threshold, the function  $\mathbf{f}$  returns  $\phi$  for the given mention. The conditions  $\mathcal{G}$  and  $\mathcal{T}$  in the functions  $\mathbf{e}$  and  $\mathbf{c}$  are omitted in the equation for simplicity.

We adopt the deep structured semantic model (DSSM) (Huang et al., 2013) as the baseline model for NED(Nie and Pan, 2018). Based on a self-attention matching network, DSSM maps the candidate entities and the context to the same semantic space, and finds the candidate entity that best semantically matches the context. The representation learning for both entities and contexts is enhanced

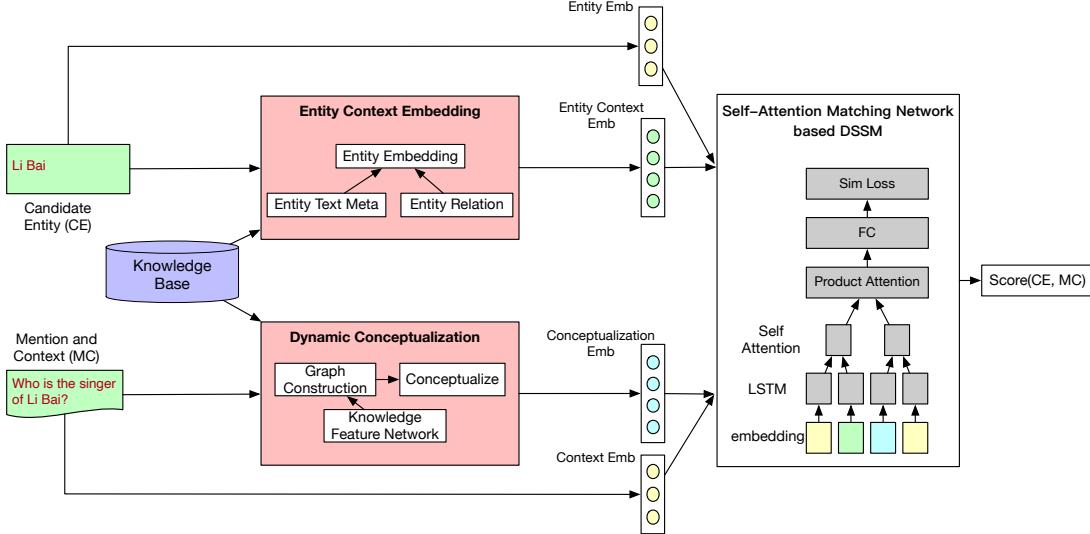


Figure 2: The architecture of our proposed entity disambiguation model K-NED.

by word2vec (Tomas Mikolov, 2013). In this work, we focus on the problem of NED itself, that is, predicting the right entity candidate for each given entity mention. The entity mentions needed for NED are simply derived from the results of named entity recognition (NER).

### 3 Knowledge-Enhanced NED Model

For short text used in information acquisition scenarios such as information retrieval and question answering, the lack of both lexical and syntactical information obstacles the precise disambiguation of entity mentions. For human beings, however, the problem of information scarcity does not hinder the disambiguation procedure. This is because there are many implicit assumptions and apriori knowledges in these information acquisition scenarios, which can be effectively considered by association and imagination during disambiguation procedure. Inspired by this, we propose a knowledge-enhanced NED model (K-NED) for short text, where two kinds of knowledge are introduce to provide additional information for better disambiguation performance. Figure 2 gives the overall architecture of the model.

The overall procedure of the K-NED model is a pipeline including feature extraction and semantic matching, where the former is composed of two sub-procedures, taking charge of feature extraction for mention context and candidate entity, respectively. Rather than considering only the utterance of the input text, the feature extraction procedure also considers external knowledge for better

representation. In details, the feature extraction sub-procedure for mention context is enhanced by conceptual knowledge formalized as a conceptual KG, which augments the representation of the mention context by giving each word a concept embedding; while the sub-procedure for candidate entity is enhanced by factual knowledge formalized as a factual KG, which augments the representation of each candidate entity by giving the entity an entity embedding. The augmented representation is used in the following semantic matching procedure for better prediction.

The major difference of the K-NED model is the introduction of external knowledge in the representation learning procedure. For the representation learning procedure, it simply uses the pre-trained word2vec language model to take charge of the conventional utterance representation learning. For the semantic matching procedure, it directly adopt the self-attention matching network based on DSSM. Given a mention  $m$  and a candidate entity  $e$ , the word2vec-based module gives two representation vectors,  $\mathbf{r}_m^{lm}$  and  $\mathbf{r}_e^{lm}$ , while the KG-based modules give another two representation vectors,  $\mathbf{r}_m^{kg}$  and  $\mathbf{r}_e^{kg}$ . The concatenation of the four representation vectors is fed into the matching network to obtain the matching degree. Based on the utterance of  $m$  or  $e$ , the word2vec-based representation vector  $\mathbf{r}_m^{lm}$  or  $\mathbf{r}_e^{lm}$  is obtained by averaging the hidden representations for the words or characters in the utterance.

We omit the detailed descriptions of the word2vec-based feature extraction and the DSSM-

based semantic matching owing to space limitations. In the following subsections, we describe in details the computation procedures for the KG-based feature extraction.

### 3.1 KG-enhanced Representation of Mention

The concepts in a conceptual KG can be treated as upper classes of the entities in the factual KG. A concept is a name or label representing a concrete or material existence such as a person, a place or a thing. For example, the entity apple, maybe corresponds to the concept of fruits, companies and songs. For a mention, we label the mention word with a concept and use the concept representation as additional feature representation of the mention. Intuitively, the concept labeling procedure works as a semantic bridge between the mentions and the entities.

Different from traditional methods where mentions are classified into coarse-grained entity types, the concept labeling procedure in our work classifies the mentions into fine-grained concepts, which can better utilize the context of the mentions and provide more information for disambiguation. We adopt a graph-based labeling algorithm for concept labeling, as shown in Figure 3. Given a short sentence, it first builds a knowledge feature network (KFN) based on the short sentence and reference conceptual/factual KGs, and then searches for the appropriate concept for the mention by a random walking algorithm. The KFN is built according to the correspondence between the symbols in the short sentence and the reference KGs. The symbols include words, entity mentions and candidate concepts, where the words and mentions are obtained by lexical analysis and entity recognition, and the concepts are obtained by matching on the reference KGs. The KFN describes three kinds of relationships, that is, the entity-concept relationship, the concept-concept relationship and the word-concept relationship.

The concept-entity relationship is represented by the generation probability from concept  $c$  to entity  $e$ . The probability  $p(c|e)$  is calculated based on the page-view (PV) statistics of the Wikipedia entity pages:

$$P(c|e) = \frac{N_{PV}(e)}{\sum_{e' \in c} N_{PV}(e')} \quad (2)$$

The concept-concept relationship is represented by the transition probability between two concepts,

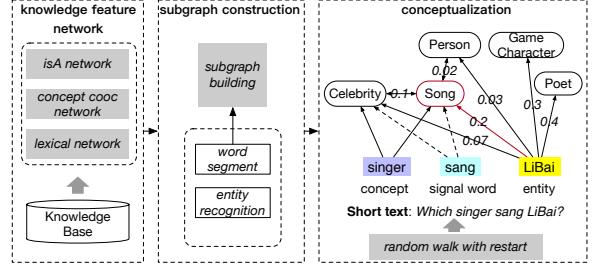


Figure 3: Architecture of fine-grained conceptualization, which consists of three parts: (a) Knowledge Feature Network. (b) Sub-graph construction. (c) Conceptualization.

$c_i$  and  $c_j$ . The probability  $P(c_i|c_j)$  is calculated based on the co-occurrence frequencies of the entities under the two concepts:

$$P(c_i|c_j) = \frac{\sum_{e_j \in c_j, e_i \in c_i} N(e_j, e_i)}{\sum_{c \in C} \sum_{e_j \in c, e_i \in c} N(e_j, e_i)} \quad (3)$$

where the co-occurrence frequency  $N(e_j, e_i)$ , is calculated based on the statistics of anchor links of Baidu Encyclopedia, and  $w$  is the size of the window that counts the co-occurrences frequencies of the entity pair in Baidu Encyclopedia. In this paper,  $w$  is set to 25.

$$N(e_j, e_i) = freq_w(e_j, e_i) \quad (4)$$

The word-concept relationship is represented by the labeling probability between the word  $w$  and the related concept  $c$ . The probability is calculated based on the word frequency and word-concept co-occurrence frequency:

$$P(c|w) = \frac{N(c, w)}{N(w)} \quad (5)$$

We perform a random walk algorithm (Jia-Yu Pan, 2004) on the KFN to get the appropriate concept of entity mention. First, we initialize the weights of the nodes and the edges by:

$$\mathbf{E}^0(e) = \begin{cases} P(c|t) & \text{if } e \text{ is } c \rightarrow t \\ P(c_i|c_j) & \text{if } e \text{ is } c_j \rightarrow c_i \end{cases} \quad (6)$$

$$\mathbf{N}^0(n) = \begin{cases} 1/|T| & \text{if } n \text{ is entity} \\ 0 & \text{if } n \text{ is concept} \end{cases} \quad (7)$$

Second, we iteratively update the node and edge by:

$$\mathbf{N}^k = (1 - \alpha)\mathbf{E}' \times \mathbf{N}^{k-1} + \alpha\mathbf{N}^0 \quad (8)$$

$$\mathbf{E}^k \leftarrow (1 - \beta)\mathbf{N}^k + \beta\mathbf{E}^k \quad (9)$$

where  $\alpha$  and  $\beta$  are hyper-parameters tuned on developing sets. Finally, we normalize the edge weights and obtain the concept type with the highest weight:

$$\begin{aligned} c^* &= \arg \max_c P(c|t) \\ &= \arg \max_c \frac{\mathbf{E}(t \rightarrow c)}{\sum_{c_i} \mathbf{E}(t \rightarrow c_i)} \end{aligned} \quad (10)$$

### 3.2 KG-enhanced Representation of Entity

The conventional representation for an entity is the textual representation of the entity. Inspired by the wide usage of distributed representation of KG entities in many NLP applications, we think that such knowledge representation is also helpful in NED. In this work, we use both textual and knowledge representation to better represent the semantics of candidate entities. We propose a novel representation learning method which can simultaneously learn both kinds of knowledge. Based on the related textual context and other information of the entities, it uses the CBOW model with a sigmoid layer to generate the distributed representation of the entities.

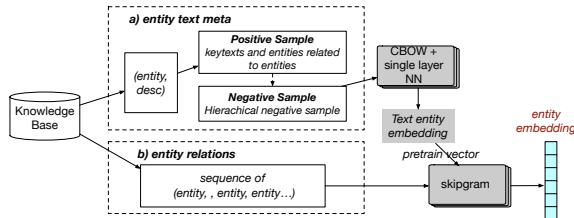


Figure 4: Entity context embedding architecture which combines entity relations and the entity context.

The detailed training process for the two models will now be introduced. Figure 4 shows that the entity  $e$  and its description generate the entity embedding. First, a positive sample is generated by entity description from KB (Wikipedia and Baidu Encyclopedia), and then word segmentation is applied to the entity description text. We have counted the word frequency in positive samples, and negative samples are generated by band-frequency random sampling. In order to learn the relationship between entities and enhanced entity representations, we use entity co-occurrence data and KB S-P-O data to generate training samples:

- entity      co-occurrence      sequence

$\{e_1, e_2, \dots, e_n\}$ , which are extracted from KB hyperlinks.

- S-P-O triples from KB, which are extracted from the key-value block of Wikipedia and Baidu Encyclopedia.

We obtained entity sequences as training sample, where each entity has an entity embedding. Then we updated the entity embedding representation with Skip-Gram Model to enhance the inter-entity relationships. Finally, we obtain as the final entity embedding. Entity embedding vector are input as feature representations of entities into an KG-enhanced entity disambiguation network, as shown in Figure 2.

## 4 Experiments and Analysis

In this section, we first introduce the experimental dataset, and construction methods of the dataset we published with this paper. Then, we present evaluation metrics, the experiments conducted for both the English and the Chinese datasets with existing approaches, and we analyze the experimental results in detail.

### 4.1 Datasets

We have experimented on both Chinese and English datasets. For the English experiment, we use Wikipedia with a release time of 202003 as KB and apply the framework to NEEL and KORE50 datasets. For the Chinese experiment, due to the lack of large-scale short text entity disambiguation datasets, we constructed a dataset called DUEL and use it as the Chinese experiment dataset alongside the FUDAN dataset ([Xu et al., 2017](#)).

#### 4.1.1 English Datasets

Most of the existing datasets on NED are based on long text, which are not suitable for our task. Two English datasets could be found that were suitable for short text entity disambiguation. Because KORE50 only has test data, but no training data, we use the training samples of NEEL as the training samples of KORE50 as well, to compare their performances.

- NEEL([Rizzo et al., 2017](#)): The training dataset consists of 6,025 tweets, the validation dataset consists of 100 tweets, and the testing dataset consists of 300 tweets.

- KORE50([Hoffart et al., 2012](#)): It contains 50 short sentences with highly ambiguous mentioned entities. It is considered to be among the most challenging for NED. Average sentence length (after removing stop words) is 6.88 words per sentence and each sentence has 2.96 mentioned entities on average.

#### 4.1.2 Chinese Datasets

The typical size of existing Chinese NED datasets is about a few thousand annotated words ([Rizzo et al., 2017](#); [Hoffart et al., 2012](#)). Because there is a lack of existing data sets for short text NED, we manually construct the largest available human annotated Chinese dataset, and we have released it to the global research community, please refer to this ([DUEL](#)) for more data details.

#### 4.1.3 Construction of Our Dataset

Our dataset provides a high-precision manually-annotated entity disambiguation dataset consisting of 100,000 short texts. The text corpus consists of queries and web page titles. The annotated entities are in the general domain, including instances (e.g. Barack Obama) and concepts (e.g. Basketball player). Table 1 and 2 depict the statistical data of the KB and the annotated text.

Table 1: Statistics of knowledge base in our dataset. AvgNumOfEntityProperties is the average number of attributes for all entities.

Statistic	KB
#Entities	398082
#SPO	3564565
#EntityDesc	361778
#AvgNumOfEntityProperties	9

**Data Annotation Method:** We annotated the entire short text in the dataset by crowd-sourcing. The same data was repeatedly labeled by three domain experts, then reviewed and released by additional experts. The average precision of annotating entities is about 95.2%. The evaluation method of dataset is as follows: given an input of a short text  $q$ , the annotated entities is  $E'_q = e'_1, e'_2, e'_3, \dots$ . By comparing the outputs  $E'_q$  with additional experts-annotated set  $E_q = e_1, e_2, e_3, \dots$ , precision  $P$  is defined as follows.

$$P = \frac{\sum_{q \in Q} E_q \cap E'_q}{\sum_{q \in Q} E_q} \quad (11)$$

**Comparison with previous datasets:** As summarized in Table 2, FUDAN is a representative evaluation dataset for Chinese short text entity disambiguation, which consists of manually annotated short text. Both FUDAN and DUEL consist of entities in various domains (including instances and concepts), such as persons, movies, and general concepts. However, DUEL is much larger.

## 4.2 Results and Analysis

### 4.2.1 Evaluation Metrics

We directly use the gold standard in mentioned entities - the NER results in the dataset, and choose standard micro F1 score as our performance metric for NED task (aggregated over all mentions).

### 4.2.2 Performance Comparison with Other Approaches

In order to verify the enhancement of different methods used in NED, we compare the proposed method with several state-of-the-art approaches both for the Chinese and the English datasets. All of these methods are effective and comparable in the case of short text. Our method is called knowledge-enhanced NED (K-NED).

- FEL([Blanco et al., 2015](#)): A toolkit for training models to link entities to KB in documents and queries. And we use DSSM model to use this entity embedding for comparing. We experiment with default parameters.
- NTEE ([Yamada et al., 2017](#)): A neural network model that learns embedding of texts and Wikipedia entities, and then use them in entity linking task. We experiment with default parameters.
- Mulrel-nel ([Le and Titov., 2018](#)): A python implementation of multi-relational NED. We experiment with default parameters.
- Fudan ([Xu et al., 2017](#)): Entity linking of Fudan University which is a Chinese entity linking service API.

As summarized in Tables 3, the experimental results indicate that our approach K-NED outperforms existing state-of-the-art methods such as FEL, NTEE, Mulrel-nil and Fudan on Chinese and English datasets except on KORE50. In particular, our method disambiguate to all correct result of the examples in Figure 1. We found that 72%

Table 2: Comparisons between DUEL and the FUDAN dataset. AvgLen is the average length of the annotated text. AvgNumEntity is the average number of entities in the annotated text.

Statistic	DUEL	FUDAN	NEEL	KORE50
#Train	90000	-	6025	-
#Dev	10000	-	100	-
#Test	10000	1037	300	50
#AvgLen	21.73	23.38	16.5157	6.88
#AvgNumEntity	3.43	2.08	2.1	2.96
#Accuracy	95.2%	-	-	-

Table 3: F1 scores on Chinese and English datasets.

Method	Datasets	
	Chinese datasets	
Fudan	DUEL 0.861	FUDAN 0.945
Mulrel-nel	0.889	0.893
K-NED(Ours)	<b>0.897</b>	<b>0.947</b>
English datasets		
FEL	NEEL 0.601	KORE50 0.360
NTEE	0.748	0.618
Mulrel-nel	0.805	<b>0.625</b>
K-NED(Ours)	<b>0.811</b>	0.544

of the types of annotated entities in the KORE50 dataset belong to the category "Person", and so it is possible that this dataset distribution is biased. Compared to KORE50, NEEL, FUDAN and DUEL datasets are more consistent with the entity type distribution of practical scenarios. NTEE and FEL use representational learning to improve performance, Mulrel-nel relied on supervised systems or heuristics to predict these relations and treat relations as latent variables in neural entity disambiguation model, and our approach uses knowledge enhancement to improve the performance without using other complex features. Data analyses demonstrate that each short text contains 3 mentioned entities on average, each of which includes 20 ambiguous entities to be linked, and the context is sparse. Experiments demonstrate that knowledge enhancement is helpful for short text entity disambiguation.

#### 4.2.3 Performance of Knowledge-Enhancement Components

In order to gain a deeper understanding of the various components of our model, we compare the difference in performance after removing two com-

ponents separately, where all models are trained using the same settings.

Table 4: F1 scores of each component on Chinese and English datasets.

Feature	DUEL	NEEL
K-NED	0.897	0.811
K-NED -DC	0.804	0.755
K-NED -ECE	0.874	0.779
K-NED -DC-ECE	0.759	0.577

As listed in Table 4, K-NED is the result of our complete model. DC represents the fine-grained dynamic conceptualization component, and ECE represents entity context embedding components. We find that dynamic conceptualization exhibits a 10.36% improvement in performance, and entity context embedding exhibits a 2.56% improvement in performance on our Chinese dataset: DUEL. By the analysis of examples in Figure 1, we find that dynamic conceptualization can mark the concepts of "Li Bai" in "Who is the singer of Li Bai?", "How to play the hero Li Bai?" and "Which famous poems are written by Li Bai?" as "songs", "game characters" and "poets" respectively. The correct conceptualization greatly facilitates the entity disambiguation. On the other hand, however, although we successfully mark the concept of "Li Bai" in "Wedding photos of Li Bai and Sa Beining" as "person", it still disambiguates incorrectly without the help of entity context embedding, which indicates that dynamic conceptualization and entity context embedding can be complementary in NED.

This result demonstrates that the conceptualization of entities is more direct and effective for the semantic disambiguation in short text entity disambiguation. In addition, we find that fine-grained conceptualization plays a significant role in dynamic conceptualization.

## 5 Related works

Many efforts have been devoted to NED in recent years. Some methods (Shen et al., 2014; Ratinov et al., 2011; Shen et al., 2012b,a; Han, 2015) exploit the Learning To Rank framework (LTR) (Liu, 2009) to rank the candidate entities, taking advantage of the relationships between all candidates. Most commonly used ranking models are the pairwise framework (Perceptron (Shen and Joshi, 2005), RankSvm(Chingpei Lee, 2014)) and the list-wise framework (ListNet (Cao et al., 2007)). (Bao-Xing et al., 2014) proposed a named entity linking method based on a probabilistic topic model(Blei, 2012), which employs the conceptual topic model to map words and mentioned entities into the same topic space. (Nakashole et al., 2013) used a graph-based collaborative entity linking model. (Bilenko et al., 2003) proposed using random walks for entity linking. Some models choosed to rely solely on the context of the links to learn entity representations, such as (Lazic et al., 2015), and some methods used a pipeline of existing annotators to filter entity candidates such as (Ling et al., 2015). Different from these conventional work, we use multiple sources of information and a deep structured semantic model to achieve better NED performance.

Many efforts have been devoted to NED for queries, such as (Hasibi et al., 2015) and (Hasibi et al., 2017). Some approaches try to solve NED by making extensive use of deep neural networks (Globerson et al., 2016), or by adopting distributed representations of words or entities (Yamada et al., 2016, 2017). Other existing approaches take advantage of global context, which captures the coherence between mapped entities of the related keywords in a document (Cucerzan, 2007; Han et al., 2011). In (Globerson et al., 2016), the neural network model uses attention mechanism to focus on the contextual entities to be disambiguated. In (Yamada et al., 2016, 2017), the distributed representation of contexts models the relationships between words and entities or between documents and entities, where the distances between various vectors provides useful information for disambiguation. Different from these work where complicated techniques or features are used, we adopt external knowledge including factual and conceptual knowledge graphs for better NED performance. (Radhakrishnan and Varma, 2018) proposes a method to train entity embedding for entity similarity, but this method relies on a dense knowledge map, we

use the text and relationship information of entity to model the similarity between entity and context to improve the effect of NED.

There are also previous work using concept or type information to improve NED performance. The models of (Hua et al., 2015; Wang et al., 2015; Priya Radhakrishnan, 2018; Isaiah Onando Muliang, 2020) try to map short text to a concept space, and then generate comprehensive concept vectors to represent the short text. (Raiman and Raiman, 2018) constructs a type ontology and a type classifier to map entities to a closed type ontology. (Chen and Xiao, 2018) proposes to modeling context explicitly by entity concept, but we use a complementary way of coarse-grained and fine-grained to dynamically predict the concept according to the context and improve the effect of disambiguation. (Derczynski et al., 2015) studies named entity recognition (NER) and named entity linking (NEL) for tweets. Unlike these work, our method uses fine-grained entity concepts and predicts concepts more accurately by using an advanced knowledge feature network.

## 6 Conclusion

We propose a knowledge-enhanced approach to short text entity disambiguation. Through bridging and facilitating semantic understanding of the fine-grained concept associated to a mentioned entity and entity embedding, the performance of entity disambiguation can be significantly improved. The experimental results demonstrate that our approach outperforms existing SOTA methods on English and Chinese datasets for this task. At the same time, we constructed a large-scale manual-annotated Chinese dataset for short text entity disambiguation, which has been released with the paper for use by researchers. As a future direction of research, we plan to explore better conceptualization and semantic understanding methods, and further improve the performance of the short text entity disambiguation task. We intend to continue to update our Chinese dataset.

In the future, We will use more modern embedding(such as BERT) or encoder(such as transformer) to obtain better embedding, and we also plan to conduct experiments to verify the effectiveness of our methods in other tasks related to semantic understanding such as Q&A, Dialogue, etc.

## References

- HUAI Bao-Xing, BAO Teng-Fei, ZHU Heng-Shu, and LIU Qi. 2014. Topic modeling approach to named entity linking. *Journal of Software*.
- M. Bilenko, R. Mooney, W. Cohen, P. Ravikumar, and S. Fien-berg. 2003. Adaptive name matching in information integration. *IEEE Intelligent Systems*.
- Roi Blanco, Giuseppe Ottaviano, and Edgar Meij. 2015. Fast and space-efficient entity linking for queries. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, pages 179–188. ACM.
- David M. Blei. 2012. *Probabilistic Topic Models*. Magazine Communications of the ACM, ACM New York, NY, USA.
- Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, and H. Li. 2007. Learning to rank: from pairwise approach to listwise approach. *ICML*.
- Liang J. Xie C. Chen, L. and Y. Xiao. 2018. Short text entity linking with fine-grained topics. pages 457–466. *CIKM*.
- Chihjen Lin Chingpei Lee. 2014. Large-scale linear ranksvm. pages 781–817. *Neural Computation*.
- S Cucerzan. 2007. Large-scale named entity disambiguation based on wikipedia data. *EMNLP CoNLL*.
- Leon Derczynski, Diana Maynard, Giuseppe Rizzo, Marieke Van Erp, Genevieve Gorrell, Raphaël Troncy, Johann Petrak, and Kalina Bontcheva. 2015. Analysis of named entity recognition and linking for tweets. *Information Processing & Management*, 51(2):32–49.
- A. Globerson, N. Lazic, S. Chakrabarti, A. Subramanya, M. Ringgaard, and F.n Pereira. 2016. Collective entity resolution with multifocal attention. *ACL*.
- Wei Shen; Jianyong Wang; Jiawei Han. 2015. Entity linking with a knowledge base: Issues, techniques, and solutions. pages 443 – 460. *IEEE Transactions on Knowledge and Data Engineering*.
- Xianpei Han, Le Sun, and Jun Zhao. 2011. Collective entity linking in web text: a graph-based method. page 765–774. *SIGIR*.
- Faegheh Hasibi, Krisztian Balog, and Svein Erik Bratsberg. 2015. Entity linking in queries: Tasks and evaluation. In *Proceedings of the 2015 International Conference on The Theory of Information Retrieval*, pages 171–180. ACM.
- Faegheh Hasibi, Krisztian Balog, and Svein Erik Bratsberg. 2017. Entity linking in queries: Efficiency vs. effectiveness. In *European Conference on Information Retrieval*, pages 40–53. Springer.
- J. Hoffart, S. Seufert, D.B. Nguyen, M. Theobald, and G. Weikum. 2012. Kore: Keyphrase overlap relatedness for entity disambiguation. pages 545–554. ACM international conference on Information and knowledge management.
- Wen Hua, Zhongyuan Wang, Haixun Wang, Kai Zheng, and Xiaofang Zhou. 2015. Short text understanding through lexical-semantic analysis. *IEEE 31st International Conference on Data Engineering*.
- Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. pages 2333–2338. *CIKM*.
- Akhilesh Vyas Saeedeh Shekarpour Maria Esther Vidal Soren Auer Jens Lehmann Isaiah Onando Mu-lang, Kuldeep Singh. 2020. Encoding knowledge graph entity aliases in attentive neural network for wikidata entity linking. page 15. *WISE*.
- Christos Faloutsos Pinar Duygulu Jia-Yu Pan, Hyung-Jeong Yang. 2004. Automatic multimedia cross-modal correlation discovery. pages 653–658. *KDD*.
- Nevena Lazic, Amarnag Subramanya, Michael Ringgaard, and Fernando Pereira. 2015. Plato: A selective context model for entity resolution. *TACL*.
- Phong Le and Ivan Titov. 2018. Improving entity linking by modeling latent relations between mentions. *ACL*.
- Xiao Ling, Sameer Singh, and Daniel S Weld. 2015. Design challenges for entity linking. *TACL*.
- T.Y. Liu. 2009. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*.
- N. Nakashole, T. Tylenda, and G. Weikum. 2013. Fine-grained semantic typing of emerging entities. *ACL*.
- Zhou S.-Liu J. Wang J. Lin C.Y. Nie, F. and R. Pan. 2018. Aggregated semantic matching for short text entity linking. pages 476–485. *CoNLL*.
- Thomas Hofmann Octavian-Eugen Ganea. 2017. Deep joint entity disambiguation with local neural attention. page 2619–2629. *ACL*.
- Vasudeva Varma Priya Radhakrishnan, Partha Talukdar. 2018. Elden: Improved entity linking using densified knowledge graphs. page 1844–1853. *NAACL*.
- Talukdar P. Radhakrishnan, P. and V. Varma. 2018. Elden: Improved entity linking using densified knowledge graphs. pages 1844–1853. *NAACL*.
- Jonathan Raiman and Olivier Raiman. 2018. Deep-type: Multilingual entity linking by neural type system evolution. *AAAI*.
- L. Ratinov, D. Roth, D. Downey, and M. Anderson. 2011. Local and global algorithms for disambiguation to wikipedia. *ACL*.

G. Rizzo, B. Pereira, A. Varga, M. van Erp M, and A.E. Cano Basave. 2017. the named entity recognition and linking (neel) challenge series. Semantic Web Journal (in press).

L. Shen and A. K. Joshi. 2005. Ranking and reranking with per-ceptron. Mach. Learn.

W. Shen, J. Wang, P. Luo, and M. Wang. 2012a. Liege: Link entities in web lists with knowledge base. SIGKDD.

W. Shen, J. Wang, P. Luo, and M. Wang. 2012b. Linden: linking named entities with knowledge base via semantic knowledge. WWW.

Wei Shen, Jianyong Wang, and Jiawei Han. 2014. Entity linking with a knowledge base: Issues, techniques, and solutions. pages 443–460. IEEE Transactions on Knowledge and Data Engineering.

Kai Chen Greg Corrado Jeffrey Dean Tomas Mikolov, Ilya Sutskever. 2013. Distributed representations of words and phrases and their compositionality. pages 3111–3119. NIPS.

Zhongyuan Wang, Kejun Zhao, Haixun Wang, Xiaofeng Meng, and Ji-Rong Wen. 2015. Query understanding through knowledge-based conceptualization. IJCAI.

Bo Xu, Yong Xu, Jiaqing Liang, Chenhao Xie, Bin Liang, Wanyun Cui, and Yanghua Xiao. 2017. Cn-dbpedia: A never-ending chinese knowledge extraction system. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, pages 428–438. Springer.

I. Yamada, H. Shindo, H. Takeda, and Y. Takefuji. 2017. Learning distributed representations of texts and entities from knowledge base. arXiv preprint.

I. Yamada, H. Shindo, and H. Takedaand Y. Takefuji. 2016. Joint learning of the embedding of words and entities for named entity disambiguation. arXiv preprint.

Luo Si Jeongwoo Ko Aditya P. Mathur Yi Fang, Naveen Somasundaram. 2011. Analysis of an expert search query log. pages 1189–1190. SIGIR.

# More Data, More Relations, More Context and More Openness: A Review and Outlook for Relation Extraction

Xu Han<sup>1\*</sup>, Tianyu Gao<sup>2\*</sup>, Yankai Lin<sup>3\*</sup>, Hao Peng<sup>1</sup>, Yaoliang Yang<sup>1</sup>, Chaojun Xiao<sup>1</sup>,  
Zhiyuan Liu<sup>1†</sup>, Peng Li<sup>3</sup>, Maosong Sun<sup>1</sup>, Jie Zhou<sup>3</sup>

<sup>1</sup>Department of Computer Science and Technology, Tsinghua University, Beijing, China

<sup>2</sup>Princeton University, Princeton, NJ, USA

<sup>3</sup>Pattern Recognition Center, WeChat AI, Tencent Inc, China

hanxu17@mails.tsinghua.edu.cn, tianyug@princeton.edu,

yankailin@tencent.com

## Abstract

Relational facts are an important component of human knowledge, which are hidden in vast amounts of text. In order to extract these facts from text, people have been working on relation extraction (RE) for years. From early pattern matching to current neural networks, existing RE methods have achieved significant progress. Yet with explosion of Web text and emergence of new relations, human knowledge is increasing drastically, and we thus require “more” from RE: a more powerful RE system that can robustly utilize more data, efficiently learn more relations, easily handle more complicated context, and flexibly generalize to more open domains. In this paper, we look back at existing RE methods, analyze key challenges we are facing nowadays, and show promising directions towards more powerful RE. We hope our view can advance this field and inspire more efforts in the community.<sup>1</sup>

## 1 Introduction

Relational facts organize knowledge of the world in a triplet format. These structured facts act as an important role of human knowledge and are explicitly or implicitly hidden in the text. For example, “Steve Jobs co-founded Apple” indicates the fact (*Apple Inc.*, founded by, *Steve Jobs*), and we can also infer the fact (*USA*, contains, *New York*) from “Hamilton made its debut in New York, USA”.

As these structured facts could benefit downstream applications, e.g., knowledge graph completion (Bordes et al., 2013; Wang et al., 2014), search engine (Xiong et al., 2017; Schlichtkrull et al., 2018) and question answering (Bordes et al., 2014; Dong et al., 2015), many efforts have been devoted

to researching **relation extraction (RE)**, which aims at extracting relational facts from plain text. More specifically, after identifying entity mentions (e.g., *USA* and *New York*) in text, the main goal of RE is to classify relations (e.g., *contains*) between these entity mentions from their context.

The pioneering explorations of RE lie in statistical approaches, such as pattern mining (Huffman, 1995; Calif and Mooney, 1997), feature-based methods (Kambhatla, 2004) and graphical models (Roth and Yih, 2002). Recently, with the development of deep learning, neural models have been widely adopted for RE (Zeng et al., 2014; Zhang et al., 2015) and achieved superior results. These RE methods have bridged the gap between unstructured text and structured knowledge, and shown their effectiveness on several public benchmarks.

Despite the success of existing RE methods, most of them still work in a simplified setting. These methods mainly focus on training models with **large amounts of human annotations** to classify two given entities **within one sentence** into **pre-defined relations**. However, the real world is much more complicated than this simple setting: (1) collecting high-quality human annotations is expensive and time-consuming, (2) many long-tail relations cannot provide large amounts of training examples, (3) most facts are expressed by long context consisting of multiple sentences, and moreover (4) using a pre-defined set to cover those relations with open-ended growth is difficult. Hence, to build an effective and robust RE system for real-world deployment, there are still some more complex scenarios to be further investigated.

In this paper, we review existing RE methods (Section 2) as well as latest RE explorations (Section 3) targeting more complex RE scenarios. Those feasible approaches leading to better RE abilities still require further efforts, and here we summarize them into four directions:

\* indicates equal contribution

† Corresponding author e-mail: liuyz@tsinghua.edu.cn

<sup>1</sup>Most of the papers mentioned in this work are collected into the following paper list <https://github.com/thunlp/NREPaper>.

(1) **Utilizing More Data** (Section 3.1). Supervised RE methods heavily rely on expensive human annotations, while distant supervision (Mintz et al., 2009) introduces more auto-labeled data to alleviate this issue. Yet distant methods bring noise examples and just utilize single sentences mentioning entity pairs, which significantly weaken extraction performance. Designing schemas to obtain high-quality and high-coverage data to train robust RE models still remains a problem to be explored.

(2) **Performing More Efficient Learning** (Section 3.2). Lots of long-tail relations only contain a handful of training examples. However, it is hard for conventional RE methods to well generalize relation patterns from limited examples like humans. Therefore, developing efficient learning schemas to make better use of limited or few-shot examples is a potential research direction.

(3) **Handling More Complicated Context** (Section 3.3). Many relational facts are expressed in complicated context (e.g. multiple sentences or even documents), while most existing RE models focus on extracting intra-sentence relations. To cover those complex facts, it is valuable to investigate RE in more complicated context.

(4) **Orienting More Open Domains** (Section 3.4). New relations emerge every day from different domains in the real world, and thus it is hard to cover all of them by hand. However, conventional RE frameworks are generally designed for pre-defined relations. Therefore, how to automatically detect undefined relations in open domains remains an open problem.

Besides the introduction of promising directions, we also point out two key challenges for existing methods: (1) **learning from text or names** (Section 4.1) and (2) **datasets towards special interests** (Section 4.2). We hope that all these contents could encourage the community to make further exploration and breakthrough towards better RE.

## 2 Background and Existing Work

Information extraction (IE) aims at extracting structural information from unstructured text, which is an important field in natural language processing (NLP). Relation extraction (RE), as an important task in IE, particularly focuses on extracting relations between entities. A complete relation extraction system consists of a named entity recognizer to identify named entities (e.g., people, organizations, locations) from text, an entity linker to link enti-

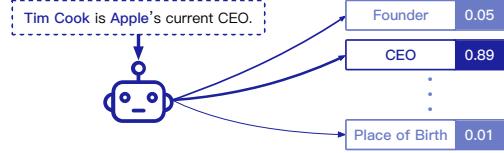


Figure 1: An example of RE. Given two entities and one sentence mentioning them, RE models classify the relation between them within a pre-defined relation set.

ties to existing knowledge graphs (KGs, necessary when using relation extraction for knowledge graph completion), and a relational classifier to determine relations between entities by given context.

Among these steps, identifying the relation is the most crucial and difficult task, since it requires models to well understand the semantics of the context. Hence, RE generally focuses on researching the classification part, which is also known as relation classification. As shown in Figure 1, a typical RE setting is that given a sentence with two marked entities, models need to classify the sentence into one of the pre-defined relations<sup>2</sup>.

In this section, we introduce the development of RE methods following the typical supervised setting, from early pattern-based methods, statistical approaches, to recent neural models.

### 2.1 Pattern Extraction Models

The pioneering methods use sentence analysis tools to identify syntactic elements in text, then automatically construct pattern rules from these elements (Soderland et al., 1995; Kim and Moldovan, 1995; Huffman, 1995; Califf and Mooney, 1997). In order to extract patterns with better coverage and accuracy, later work involves larger corpora (Carlson et al., 2010), more formats of patterns (Nakashole et al., 2012; Jiang et al., 2017), and more efficient ways of extraction (Zheng et al., 2019). As automatically constructed patterns may have mistakes, most of the above methods require further examinations from human experts, which is the main limitation of pattern-based models.

### 2.2 Statistical Relation Extraction Models

As compared to using pattern rules, statistical methods bring better coverage and require less human efforts. Thus statistical relation extraction (SRE) has been extensively studied.

<sup>2</sup>Sometimes there is a special class in the relation set indicating that the sentence does not express any pre-specified relation (usually named as N/A).

One typical SRE approach is **feature-based methods** (Kambhatla, 2004; Zhou et al., 2005; Jiang and Zhai, 2007; Nguyen et al., 2007), which design lexical, syntactic and semantic features for entity pairs and their corresponding context, and then input these features into relation classifiers.

Due to the wide use of support vector machines (SVM), **kernel-based methods** have been widely explored, which design kernel functions for SVM to measure the similarities between relation representations and textual instances (Culotta and Sorensen, 2004; Bunescu and Mooney, 2005; Zhao and Grishman, 2005; Mooney and Bunescu, 2006; Zhang et al., 2006b,a; Wang, 2008).

There are also some other statistical methods focusing on extracting and inferring the latent information hidden in the text. **Graphical methods** (Roth and Yih, 2002, 2004; Sarawagi and Cohen, 2005; Yu and Lam, 2010) abstract the dependencies between entities, text and relations in the form of directed acyclic graphs, and then use inference models to identify the correct relations.

Inspired by the success of **embedding models** in other NLP tasks (Mikolov et al., 2013a,b), there are also efforts in encoding text into low-dimensional semantic spaces and extracting relations from textual embeddings (Weston et al., 2013; Riedel et al., 2013; Gormley et al., 2015). Furthermore, Bordes et al. (2013), Wang et al. (2014) and Lin et al. (2015) utilize KG embeddings for RE.

Although SRE has been widely studied, it still faces some challenges. Feature-based and kernel-based models require many efforts to design features or kernel functions. While graphical and embedding methods can predict relations without too much human intervention, they are still limited in model capacities. There are some surveys systematically introducing SRE models (Zelenko et al., 2003; Bach and Badaskar, 2007; Pawar et al., 2017). In this paper, we do not spend too much space for SRE and focus more on neural-based models.

### 2.3 Neural Relation Extraction Models

Neural relation extraction (NRE) models introduce neural networks to automatically extract semantic features from text. Compared with SRE models, NRE methods can effectively capture textual information and generalize to wider range of data.

Studies in NRE mainly focus on designing and utilizing various network architectures to capture the relational semantics within text, such as **recurrent neural networks** (Socher et al., 2012; Miwa and Bansal, 2016) that learn compositional representations for sentences recursively, **convolutional neural networks (CNNs)** (Liu et al., 2013; Zeng et al., 2014; Santos et al., 2015; Nguyen and Grishman, 2015b; Zeng et al., 2015; Huang and Wang, 2017) that effectively model local textual patterns, **recurrent neural networks (RNNs)** (Zhang and Wang, 2015; Nguyen and Grishman, 2015a; Vu et al., 2016; Zhang et al., 2015) that can better handle long sequential data, **graph neural networks (GNNs)** (Zhang et al., 2018; Zhu et al., 2019a) that build word/entity graphs for reasoning, and **attention-based neural networks** (Zhou et al., 2016; Wang et al., 2016; Xiao and Liu, 2016) that utilize attention mechanism to aggregate global relational information.

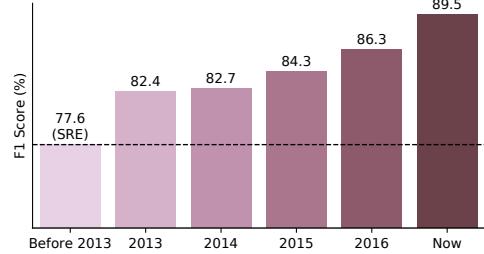


Figure 2: The performance of state-of-the-art RE models in different years on widely-used dataset SemEval-2010 Task 8. The adoption of neural models (since 2013) has brought great improvement in performance.

Different from SRE models, NRE mainly utilizes word embeddings and position embeddings instead of hand-craft features as inputs. **Word embeddings** (Turian et al., 2010; Mikolov et al., 2013b) are the most used input representations in NLP, which encode the semantic meaning of words into vectors. In order to capture the entity information in text, **position embeddings** (Zeng et al., 2014) are introduced to specify the relative distances between words and entities. Except for word embeddings and position embeddings, there are also other works integrating syntactic information into NRE models. Xu et al. (2015a) and Xu et al. (2015b) adopt CNNs and RNNs over **shortest dependency paths** respectively. Liu et al. (2015) propose a recursive neural network based on augmented dependency paths. Xu et al. (2016) and Cai et al. (2016) utilize deep RNNs to make further use of dependency paths. Besides, there are some efforts combining NRE with **universal schemas** (Verga et al., 2016; Verga and McCallum,

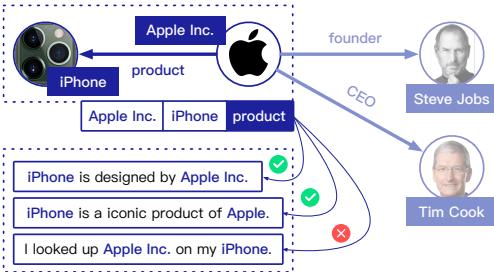


Figure 3: An example of distantly supervised relation extraction. With the fact (*Apple Inc.*, *product*, *iPhone*), DS finds all sentences mentioning the two entities and annotates them with the relation *product*, which inevitably brings noise labels.

2016; Riedel et al., 2013). Recently, **Transformers** (Vaswani et al., 2017) and **pre-trained language models** (Devlin et al., 2019) have also been explored for NRE (Du et al., 2018; Verga et al., 2018; Wu and He, 2019; Baldini Soares et al., 2019) and have achieved new state-of-the-arts.

By concisely reviewing the above techniques, we are able to track the development of RE from pattern and statistical methods to neural models. Comparing the performance of state-of-the-art RE models in years (Figure 2), we can see the vast increase since the emergence of NRE, which demonstrates the power of neural methods.

### 3 “More” Directions for RE

Although the above-mentioned NRE models have achieved superior results on benchmarks, they are still far from solving the problem of RE. Most of these models utilize abundant human annotations and just aim at extracting pre-defined relations within single sentences. Hence, it is hard for them to work well in complex cases. In fact, there have been various works exploring feasible approaches that lead to better RE abilities on real-world scenarios. In this section, we summarize these exploratory efforts into four directions, and give our review and outlook about these directions.

#### 3.1 Utilizing More Data

Supervised NRE models suffer from the lack of large-scale high-quality training data, since manually labeling data is time-consuming and human-intensive. To alleviate this issue, distant supervision (DS) assumption has been used to automatically label data by aligning existing KGs with plain text (Mintz et al., 2009; Nguyen and Moschitti, 2011; Min et al., 2013). As shown in Figure 3, for

Dataset	#Rel.	#Fact	#Inst.	N/A
NYT-10	53	377,980	694,491	79.43%
Wiki-Distant	454	605,877	1,108,288	47.61%

Table 1: Statistics for NYT-10 and Wiki-Distant. Four columns stand for numbers of relations, facts and instances, and proportions of N/A instances respectively.

Model	NYT-10	Wiki-Distant
PCNN-ONE	0.340	0.214
PCNN-ATT	0.349	0.222
BERT	0.458	0.361

Table 2: Area under the curve (AUC) of PCNN-ONE (Zeng et al., 2015), PCNN-ATT (Lin et al., 2016) and BERT (Devlin et al., 2019) on two datasets.

any entity pair in KGs, sentences mentioning both the entities will be labeled with their corresponding relations in KGs. Large-scale training examples can be easily constructed by this heuristic scheme.

Although DS provides a feasible approach to utilize more data, this automatic labeling mechanism is inevitably accompanied by the wrong labeling problem. The reason is that not all sentences mentioning the two entities express their relations in KGs exactly. For example, we may mistakenly label “*Bill Gates* retired from *Microsoft*” with the relation *founder*, if (*Bill Gates*, *founder*, *Microsoft*) is a relational fact in KGs.

The existing methods to alleviate the noise problem can be divided into three major approaches:

(1) Some methods adopt multi-instance learning by combining sentences with same entity pairs and then **selecting informative instances** from them. Riedel et al. (2010); Hoffmann et al. (2011); Surdeanu et al. (2012) utilize graphical model to infer the informative sentences, while Zeng et al. (2015) use a simple heuristic selection strategy. Later on, Lin et al. (2016); Zhang et al. (2017); Han et al. (2018c); Li et al. (2020); Zhu et al. (2019c); Hu et al. (2019) design attention mechanisms to highlight informative instances for RE.

(2) **Incorporating extra context information** to denoise DS data has also been explored, such as incorporating KGs as external information to guide instance selection (Ji et al., 2017; Han et al., 2018b; Zhang et al., 2019a; Qu et al., 2019) and adopting multi-lingual corpora for the information consistency and complementarity (Verga et al., 2016; Lin et al., 2017; Wang et al., 2018).

(3) Many methods tend to utilize **sophisticated**

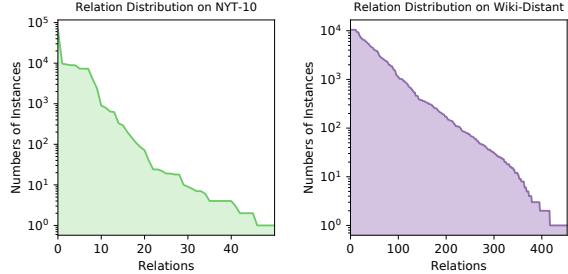


Figure 4: Relation distributions (log-scale) on the training part of DS datasets NYT-10 and Wiki-Distant, suggesting that real-world relation distributions suffer from the long-tail problem.

**mechanisms and training strategies** to enhance distantly supervised NRE models. Vu et al. (2016); Beltagy et al. (2019) combine different architectures and training strategies to construct hybrid frameworks. Liu et al. (2017) incorporate a soft-label scheme by changing unconfident labels during training. Furthermore, reinforcement learning (Feng et al., 2018; Zeng et al., 2018) and adversarial training (Wu et al., 2017; Wang et al., 2018; Han et al., 2018a) have also been adopted in DS.

The researchers have formed a consensus that utilizing more data is a potential way towards more powerful RE models, and there still remains some open problems worth exploring:

(1) Existing DS methods focus on **denoising auto-labeled instances** and it is certainly meaningful to follow this research direction. Besides, current DS schemes are still similar to the original one in (Mintz et al., 2009), which just covers the case that the entity pairs are mentioned in the same sentences. To achieve better coverage and less noise, **exploring better DS schemes** for auto-labeling data is also valuable.

(2) Inspired by recent work in adopting pre-trained language models (Zhang et al., 2019b; Wu and He, 2019; Baldini Soares et al., 2019) and active learning (Zheng et al., 2019) for RE, to **perform unsupervised or semi-supervised learning** for utilizing large-scale unlabeled data as well as using knowledge from KGs and introducing human experts in the loop is also promising.

Besides addressing existing approaches and future directions, we also propose a new DS dataset to advance this field, which will be released once the paper is published. The most used benchmark for DS, NYT-10 (Riedel et al., 2010), suffers from small amount of relations, limited relation domains and extreme long-tail relation performance. To

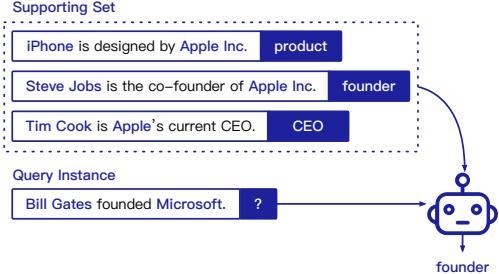


Figure 5: An example of few-shot RE. Give a few instances for new relation types, few-shot RE models classify query sentences into one of the given relations.

alleviate these drawbacks, we utilize Wikipedia and Wikidata (Vrandečić and Krötzsch, 2014) to construct **Wiki-Distant** in the same way as Riedel et al. (2010). As demonstrated in Table 1, Wiki-Distant covers more relations and possesses more instances, with a more reasonable N/A proportion. Comparison results of state-of-the-art models on these two datasets<sup>3</sup> are shown in Table 2, indicating that Wiki-Distant is more challenging and there is a long way to resolve distantly supervised RE.

### 3.2 Performing More Efficient Learning

Real-world relation distributions are long-tail: Only the common relations obtain sufficient training instances and most relations have very limited relational facts and corresponding sentences. We can see the long-tail relation distributions on two DS datasets from Figure 4, where many relations even have less than 10 training instances. This phenomenon calls for models that can learn long-tail relations more efficiently. Few-shot learning, which focuses on grasping tasks with only a few training examples, is a good fit for this need.

To advance this field, Han et al. (2018d) first built a large-scale few-shot relation extraction dataset (FewRel). This benchmark takes the  $N$ -way  $K$ -shot setting, where models are given  $N$  random-sampled new relations, along with  $K$  training examples for each relation. With limited information, RE models are required to classify query instances into given relations (Figure 5).

The general idea of few-shot models is to train good representations of instances or learn ways of fast adaptation from existing large-scale data, and then transfer to new tasks. There are mainly two ways for handling few-shot learning: (1) **Metric learning** learns a semantic metric on existing

<sup>3</sup>Due to the large size, we do not use any denoise mechanism for BERT, which still achieves the best results.

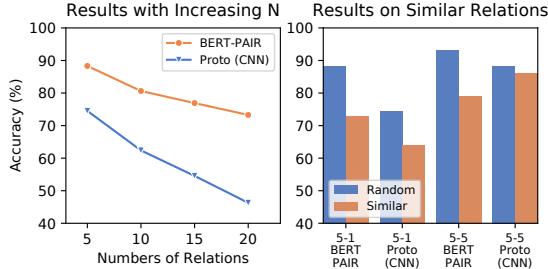


Figure 6: Few-shot RE results with (A) increasing  $N$  and (B) similar relations. The left figure shows the accuracy (%) of two models in  $N$ -way 1-shot RE. In the right figure, “random” stands for the standard few-shot setting and “similar” stands for evaluating with selected similar relations.

data and classifies queries by comparing them with training examples (Koch et al., 2015; Vinyals et al., 2016; Snell et al., 2017; Baldini Soares et al., 2019). While most metric learning models perform distance measurement on sentence-level representation, Ye and Ling (2019); Gao et al. (2019) utilize token-level attention for finer-grained comparison. (2) **Meta-learning**, also known as “learning to learn”, aims at grasping the way of parameter initialization and optimization through the experience gained on the meta-train data (Ravi and Larochelle, 2017; Finn et al., 2017; Mishra et al., 2018).

Researchers have made great progress in few-shot RE. However, there remain many challenges that are important for its applications and have not yet been discussed. Gao et al. (2019) propose two problems worth further investigation:

(1) **Few-shot domain adaptation** studies how few-shot models can transfer across domains. It is argued that in the real-world application, the test domains are typically lacking annotations and could differ vastly from the training domains. Thus, it is crucial to evaluate the transferabilities of few-shot models across domains.

(2) **Few-shot none-of-the-above detection** is about detecting query instances that do not belong to any of the sampled  $N$  relations. In the  $N$ -way  $K$ -shot setting, it is assumed that all queries express one of the given relations. However, the real case is that most sentences are not related to the relations of our interest. Conventional few-shot models cannot well handle this problem due to the difficulty to form a good representation for the none-of-the-above (NOTA) relation. Therefore, it is crucial to study how to identify NOTA instances.

(3) Besides the above challenges, it is also impor-

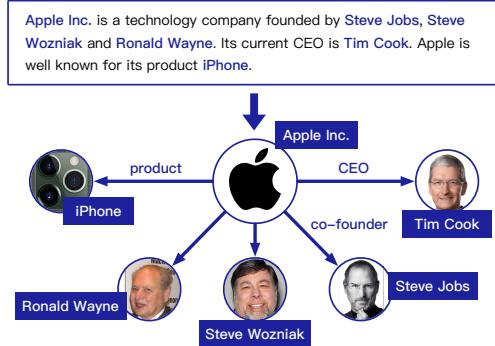


Figure 7: An example of document-level RE. Given a paragraph with several sentences and multiple entities, models are required to extract all possible relations between these entities expressed in the document.

tant to see that, the existing **evaluation protocol** may over-estimate the progress we made on few-shot RE. Unlike conventional RE tasks, few-shot RE randomly samples  $N$  relations for each evaluation episode; in this setting, the number of relations is usually very small (5 or 10) and it is very likely to sample  $N$  distinct relations and thus reduce to a very easy classification task.

We carry out two simple experiments to show the problems (Figure 6): (A) We evaluate few-shot models with increasing  $N$  and the performance drops drastically with larger relation numbers. Considering that the real-world case contains much more relations, it shows that existing models are still far from being applied. (B) Instead of randomly sampling  $N$  relations, we hand-pick 5 relations similar in semantics and evaluate few-shot RE models on them. It is no surprise to observe a sharp decrease in the results, which suggests that existing few-shot models may overfit simple textual cues between relations instead of really understanding the semantics of the context. More details about the experiments are in Appendix A.

### 3.3 Handling More Complicated Context

As shown in Figure 7, one document generally mentions many entities exhibiting complex cross-sentence relations. Most existing methods focus on intra-sentence RE and thus are inadequate for collectively identifying these relational facts expressed in a long paragraph. In fact, most relational facts can only be extracted from complicated context like documents rather than single sentences (Yao et al., 2019), which should not be neglected.

There are already some works proposed to extract relations across multiple sentences:

(1) **Syntactic methods** (Wick et al., 2006; Gerber and Chai, 2010; Swampillai and Stevenson, 2011; Yoshikawa et al., 2011; Quirk and Poon, 2017) rely on textual features extracted from various syntactic structures, such as coreference annotations, dependency parsing trees and discourse relations, to connect sentences in documents.

(2) Zeng et al. (2017); Christopoulou et al. (2018) build inter-sentence entity graphs, which can utilize **multi-hop paths between entities** for inferring the correct relations.

(3) Peng et al. (2017); Song et al. (2018); Zhu et al. (2019b) employ **graph-structured neural networks** to model cross-sentence dependencies for relation extraction, which bring in memory and reasoning abilities.

To advance this field, some document-level RE datasets have been proposed. Quirk and Poon (2017); Peng et al. (2017) build datasets by DS. Li et al. (2016); Peng et al. (2017) propose datasets for specific domains. Yao et al. (2019) construct a general document-level RE dataset annotated by crowdsourcing workers, suitable for evaluating general-purpose document-level RE systems.

Although there are some efforts investing into extracting relations from complicated context (e.g., documents), the current RE models for this challenge are still crude and straightforward. Followings are some directions worth further investigation:

(1) Extracting relations from complicated context is a challenging task requiring **reading, memorizing and reasoning** for discovering relational facts across multiple sentences. Most of current RE models are still very weak in these abilities.

(2) Besides documents, **more forms of context** is also worth exploring, such as extracting relational facts across documents, or understanding relational information based on heterogeneous data.

(3) Inspired by Narasimhan et al. (2016), which utilizes search engines for acquiring external information, **automatically searching and analysing context for RE** may help RE models identify relational facts with more coverage and become practical for daily scenarios.

### 3.4 Orienting More Open Domains

Most RE systems work within pre-specified relation sets designed by human experts. However, our world undergoes open-ended growth of relations and it is not possible to handle all these emerging



Figure 8: An example of open information extraction, which extracts relation arguments (entities) and phrases without relying on any pre-defined relation types.

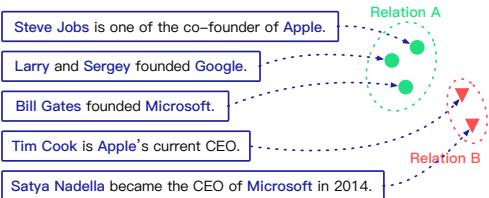


Figure 9: An example of clustering-based relation discovery, which identifying potential relation types by clustering unlabeled relational instances.

relation types only by humans. Thus, we need RE systems that do not rely on pre-defined relation schemas and can work in open scenarios.

There are already some explorations in handling open relations: (1) **Open information extraction (Open IE)**, as shown in Figure 8, extracts relation phrases and arguments (entities) from text (Banko et al., 2007; Fader et al., 2011; Mausam et al., 2012; Del Corro and Gemulla, 2013; Angeli et al., 2015; Stanovsky and Dagan, 2016; Mausam, 2016; Cui et al., 2018). Open IE does not rely on specific relation types and thus can handle all kinds of relational facts. (2) **Relation discovery**, as shown in Figure 9, aims at discovering unseen relation types from unsupervised data. Yao et al. (2011); Marcheggiani and Titov (2016) propose to use generative models and treat these relations as latent variables, while Shinyama and Sekine (2006); El-sahar et al. (2017); Wu et al. (2019) cast relation discovery as a clustering task.

Though relation extraction in open domains has been widely studied, there are still lots of unsolved research questions remained to be answered:

(1) **Canonicalizing relation phrases and arguments** in Open IE is crucial for downstream tasks (Niklaus et al., 2018). If not canonicalized, the extracted relational facts could be redundant and ambiguous. For example, Open IE may extract two triples (*Barack Obama*, was born in, *Honolulu*) and (*Obama*, place of birth, *Honolulu*) indicating an identical fact. Thus, normalizing extracted results will largely benefit the applications of Open IE. There are already some preliminary works in this area (Galárraga et al., 2014;

Vashishth et al., 2018) and more efforts are needed.

(2) The **not applicable (N/A) relation** has been hardly addressed in relation discovery. In previous work, it is usually assumed that the sentence always expresses a relation between the two entities (Marcheggiani and Titov, 2016). However, in the real-world scenario, a large proportion of entity pairs appearing in a sentence do not have a relation, and ignoring them or using simple heuristics to get rid of them may lead to poor results. Thus, it would be of interest to study how to handle these N/A instances in relation discovery.

## 4 Other Challenges

In this section, we analyze two key challenges faced by RE models, address them with experiments and show their significance in the research and development of RE systems<sup>4</sup>.

### 4.1 Learning from Text or Names

In the process of RE, both entity names and their context provide useful information for classification. **Entity names** provide typing information (e.g., we can easily tell *JFK International Airport* is an airport) and help to narrow down the range of possible relations; In the training process, entity embeddings may also be formed to help relation classification (like in the link prediction task of KG). On the other hand, relations can usually be extracted from the semantics of **text** around entity pairs. In some cases, relations can only be inferred implicitly by reasoning over the context.

Since there are two sources of information, it is interesting to study how much each of them contributes to the RE performance. Therefore, we design three different settings for the experiments: (1) **normal** setting, where both names and text are taken as inputs; (2) **masked-entity** (ME) setting, where entity names are replaced with a special token; (3) **only-entity** (OE) setting, where only names of the two entities are provided.

Results from Table 3 show that compared to the normal setting, models suffer a huge performance drop in both the ME and OE settings. Besides, it is surprising to see that in some cases, only using entity names outperforms only using text with entities masked. It suggests that (1) both entity names and text provide crucial information for RE, and

<sup>4</sup>For more details about these experiments, please refer to our open-source toolkit <https://github.com/thunlp/OpenNRE>.

Benchmark	Normal	ME	OE
Wiki80 (Acc)	0.861	0.734	0.763
TACRED (F-1)	0.666	0.554	0.412
NYT-10 (AUC)	0.349	0.216	0.185
Wiki-Distant (AUC)	0.222	0.145	0.173

Table 3: Results of state-of-the-arts models on the normal setting, masked-entity (ME) setting and only-entity (OE) setting. We report accuracies of BERT on Wiki80, F-1 scores of BERT on TACRED and AUC of PCNN-ATT on NYT-10 and Wiki-Distant. All models are from the OpenNRE package (Han et al., 2019).

(2) for some existing state-of-the-art models and benchmarks, entity names contribute even more.

The observation is contrary to human intuition: we classify the relations between given entities mainly from the text description, yet models learn more from their names. To make real progress in understanding how language expresses relational facts, this problem should be further investigated and more efforts are needed.

### 4.2 RE Datasets towards Special Interests

There are already many datasets that benefit RE research: For supervised RE, there are MUC (Grishman and Sundheim, 1996), ACE-2005 (Ntroduction, 2005), SemEval-2010 Task 8 (Hendrickx et al., 2009), KBP37 (Zhang and Wang, 2015) and TACRED (Zhang et al., 2017); and we have NYT-10 (Riedel et al., 2010), FewRel (Han et al., 2018d) and DocRED (Yao et al., 2019) for distant supervision, few-shot and document-level RE respectively.

However, there are barely datasets targeting special problems of interest. For example, RE across sentences (e.g., two entities are mentioned in two different sentences) is an important problem, yet there is no specific datasets that can help researchers study it. Though existing document-level RE datasets contain instances of this case, it is hard to analyze the exact performance gain towards this specific aspect. Usually, researchers (1) use hand-crafted sub-sets of general datasets or (2) carry out case studies to show the effectiveness of their models in specific problems, which is lacking of convincing and quantitative analysis. Therefore, to further study these problems of great importance in the development of RE, it is necessary for the community to construct well-recognized, well-designed and fine-grained datasets towards special interests.

## 5 Conclusion

In this paper, we give a comprehensive and detailed review on the development of relation extraction models, generalize four promising directions leading to more powerful RE systems (utilizing more data, performing more efficient learning, handling more complicated context and orienting more open domains), and further investigate two key challenges faced by existing RE models. We thoroughly survey the previous RE literature as well as supporting our points with statistics and experiments. Through this paper, we hope to demonstrate the progress and problems in existing RE research and encourage more efforts in this area.

## Acknowledgments

This work is supported by the Natural Science Foundation of China (NSFC) and the German Research Foundation (DFG) in Project Crossmodal Learning, NSFC 61621136008 / DFG TRR-169, and Beijing Academy of Artificial Intelligence (BAAI). This work is also supported by the Pattern Recognition Center, WeChat AI, Tencent Inc. Gao is supported by 2019 Tencent Rhino-Bird Elite Training Program. Gao is also supported by Tsinghua University Initiative Scientific Research Program.

## References

- Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D. Manning. 2015. **Leveraging linguistic structure for open domain information extraction**. In *Proceedings of ACL-IJCNLP*, pages 344–354.
- Nguyen Bach and Sameer Badaskar. 2007. **A review of relation extraction**.
- Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. **Matching the blanks: Distributional similarity for relation learning**. In *Proceedings of ACL*, pages 2895–2905.
- Michele Banko, Michael J Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. 2007. **Open information extraction from the web**. In *Proceedings of IJCAI*, pages 2670–2676.
- Iz Beltagy, Kyle Lo, and Waleed Ammar. 2019. **Combining distant and direct supervision for neural relation extraction**. In *Proceedings of NAACL-HLT*, pages 1858–1867.
- Antoine Bordes, Sumit Chopra, and Jason Weston. 2014. **Question answering with subgraph embeddings**. In *Proceedings of EMNLP*, pages 615–620.
- Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. **Translating embeddings for modeling multi-relational data**. In *Proceedings of NIPS*, pages 2787–2795.
- Razvan C Bunescu and Raymond J Mooney. 2005. **A shortest path dependency kernel for relation extraction**. In *Proceedings of EMNLP*, pages 724–731.
- Rui Cai, Xiaodong Zhang, and Houfeng Wang. 2016. **Bidirectional recurrent convolutional neural network for relation classification**. In *Proceedings of ACL*, pages 756–765.
- Mary Elaine Califf and Raymond J. Mooney. 1997. **Relational learning of pattern-match rules for information extraction**. In *Proceedings of CoNLL*, pages 9–15.
- Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka, and Tom M Mitchell. 2010. **Toward an architecture for never-ending language learning**. In *Proceedings of AAAI*, pages 1306–1313.
- Fenia Christopoulou, Makoto Miwa, and Sophia Ananiadou. 2018. **A walk-based model on entity graphs for relation extraction**. In *Proceedings of ACL*, pages 81–88.
- Lei Cui, Furu Wei, and Ming Zhou. 2018. **Neural open information extraction**. In *Proceedings of ACL*, pages 407–413.
- Aron Culotta and Jeffrey Sorensen. 2004. **Dependency tree kernels for relation extraction**. In *Proceedings of ACL*, page 423.
- Luciano Del Corro and Rainer Gemulla. 2013. **Clausie: clause-based open information extraction**. In *Proceedings of WWW*, pages 355–366.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. **Bert: Pre-training of deep bidirectional transformers for language understanding**. In *Proceedings of NAACL-HLT*, pages 4171–4186.
- Li Dong, Furu Wei, Ming Zhou, and Ke Xu. 2015. **Question answering over freebase with multi-column convolutional neural networks**. In *Proceedings of ACL-IJCNLP*, pages 260–269.
- Jinhua Du, Jingguang Han, Andy Way, and Dadong Wan. 2018. **Multi-level structured self-attentions for distantly supervised relation extraction**. In *Proceedings of EMNLP*, pages 2216–2225.
- Hady Elsayar, Elena Demidova, Simon Gottschalk, Christophe Gravier, and Frederique Laforest. 2017. **Unsupervised open relation extraction**. In *Proceedings of ESWC*, pages 12–16.

- Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. *Identifying relations for open information extraction*. In *Proceedings of EMNLP*, pages 1535–1545.
- Jun Feng, Minlie Huang, Li Zhao, Yang Yang, and Xiaoyan Zhu. 2018. *Reinforcement learning for relation classification from noisy data*. In *Proceedings of AAAI*, pages 5779–5786.
- Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. *Model-agnostic meta-learning for fast adaptation of deep networks*. In *Proceedings of ICML*, pages 1126–1135.
- Luis Galárraga, Jeremy Heitz, Kevin Murphy, and Fabian M Suchanek. 2014. *Canonicalizing open knowledge bases*. In *Proceedings of CIKM*, pages 1679–1688.
- Tianyu Gao, Xu Han, Hao Zhu, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2019. *FewRel 2.0: Towards more challenging few-shot relation classification*. In *Proceedings of EMNLP-IJCNLP*, pages 6251–6256.
- Matthew Gerber and Joyce Chai. 2010. *Beyond NomBank: A study of implicit arguments for nominal predicates*. In *Proceedings of ACL*, pages 1583–1592.
- Matthew R Gormley, Mo Yu, and Mark Dredze. 2015. *Improved relation extraction with feature-rich compositional embedding models*. In *Proceedings of EMNLP*, pages 1774–1784.
- Ralph Grishman and Beth Sundheim. 1996. *Message understanding conference- 6: A brief history*. In *Proceedings of COLING*, pages 466–471.
- Xu Han, Tianyu Gao, Yuan Yao, Deming Ye, Zhiyuan Liu, and Maosong Sun. 2019. *OpenNRE: An open and extensible toolkit for neural relation extraction*. In *Proceedings of EMNLP-IJCNLP*, pages 169–174.
- Xu Han, Zhiyuan Liu, and Maosong Sun. 2018a. *De-noising distant supervision for relation extraction via instance-level adversarial training*. *arXiv preprint arXiv:1805.10959*.
- Xu Han, Zhiyuan Liu, and Maosong Sun. 2018b. *Neural knowledge acquisition via mutual attention between knowledge graph and text*. In *Proceedings of AAAI*, pages 4832–4839.
- Xu Han, Pengfei Yu, Zhiyuan Liu, Maosong Sun, and Peng Li. 2018c. *Hierarchical relation extraction with coarse-to-fine grained attention*. In *Proceedings of EMNLP*, pages 2236–2245.
- Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018d. *Fewrel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation*. In *Proceedings of EMNLP*, pages 4803–4809.
- Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2009. *SemEval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals*. In *Proceedings of SEW-2009*, pages 94–99.
- Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S Weld. 2011. *Knowledge-based weak supervision for information extraction of overlapping relations*. In *Proceedings of ACL*, pages 541–550.
- Linmei Hu, Luhao Zhang, Chuan Shi, Liqiang Nie, Weili Guan, and Cheng Yang. 2019. *Improving distantly-supervised relation extraction with joint label embedding*. In *Proceedings of EMNLP-IJCNLP*, pages 3812–3820.
- Yi Yao Huang and William Yang Wang. 2017. *Deep residual learning for weakly-supervised relation extraction*. In *Proceedings of EMNLP*, pages 1803–1807.
- Scott B Huffman. 1995. *Learning information extraction patterns from examples*. In *Proceedings of IJCAI*, pages 246–260.
- Guoliang Ji, Kang Liu, Shizhu He, Jun Zhao, et al. 2017. *Distant supervision for relation extraction with sentence-level attention and entity descriptions*. In *AAAI*, pages 3060–3066.
- Jing Jiang and ChengXiang Zhai. 2007. *A systematic exploration of the feature space for relation extraction*. In *Proceedings of NAACL*, pages 113–120.
- Meng Jiang, Jingbo Shang, Taylor Cassidy, Xiang Ren, Lance M Kaplan, Timothy P Hanratty, and Jiawei Han. 2017. *Metapad: Meta pattern discovery from massive text corpora*. In *Proceedings of KDD*, pages 877–886.
- Nanda Kambhatla. 2004. *Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations*. In *Proceedings of ACL*, pages 178–181.
- Jun-Tae Kim and Dan I. Moldovan. 1995. *Acquisition of linguistic patterns for knowledge-based information extraction*. *TKDE*, 7(5):713–724.
- Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. 2015. *Siamese neural networks for one-shot image recognition*. In *Proceedings of the Workshop of ICML*.
- Jiao Li, Yueping Sun, Robin J. Johnson, Daniela Sciaky, Chih-Hsuan Wei, Robert Leaman, Allan Peter Davis, Carolyn J. Mattingly, Thomas C. Wiegers, and Zhiyong Lu. 2016. *BioCreative V CDR task corpus: a resource for chemical disease relation extraction*. *Database*, pages 1–10.

- Yang Li, Guodong Long, Tao Shen, Tianyi Zhou, Lina Yao, Huan Huo, and Jing Jiang. 2020. *Self-attention enhanced selective gate with entity-aware embedding for distantly supervised relation extraction*. In *Proceedings of AAAI*, pages 8269–8276.
- Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2017. *Neural relation extraction with multi-lingual attention*. In *Proceedings of ACL*, pages 34–43.
- Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. *Learning entity and relation embeddings for knowledge graph completion*. In *Proceedings of AAAI*, pages 2181–2187.
- Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. *Neural relation extraction with selective attention over instances*. In *Proceedings of ACL*, pages 2124–2133.
- Chunyang Liu, Wenbo Sun, Wenhan Chao, and Wanxiang Che. 2013. *Convolution neural network for relation extraction*. In *Proceedings of ICDM*, pages 231–242.
- Tianyu Liu, Kexiang Wang, Baobao Chang, and Zhi-fang Sui. 2017. *A soft-label method for noise-tolerant distantly supervised relation extraction*. In *Proceedings of EMNLP*, pages 1790–1795.
- Yang Liu, Furu Wei, Sujian Li, Heng Ji, Ming Zhou, and WANG Houfeng. 2015. *A dependency-based neural network for relation classification*. In *Proceedings of ACL-IJCNLP*, pages 285–290.
- Diego Marcheggiani and Ivan Titov. 2016. *Discrete-state variational autoencoders for joint discovery and factorization of relations*. *TACL*, 4:231–244.
- Mausam, Michael Schmitz, Stephen Soderland, Robert Bart, and Oren Etzioni. 2012. *Open language learning for information extraction*. In *Proceedings of EMNLP-CoNLL*, pages 523–534.
- Mausam Mausam. 2016. *Open information extraction systems and downstream applications*. In *Proceedings of IJCAI*, pages 4074–4077.
- Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. *Efficient estimation of word representations in vector space*. In *Proceedings of ICLR*.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. *Distributed representations of words and phrases and their compositionality*. In *Proceedings of NIPS*, pages 3111–3119.
- Bonan Min, Ralph Grishman, Li Wan, Chang Wang, and David Gondek. 2013. *Distant supervision for relation extraction with an incomplete knowledge base*. In *Proceedings of NAACL*, pages 777–782.
- Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. *Distant supervision for relation extraction without labeled data*. In *Proceedings of ACL-IJCNLP*, pages 1003–1011.
- Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. 2018. *A simple neural attentive meta-learner*. In *Proceedings of ICLR*.
- Makoto Miwa and Mohit Bansal. 2016. *End-to-end relation extraction using lstms on sequences and tree structures*. In *Proceedings of ACL*, pages 1105–1116.
- Raymond J Mooney and Razvan C Bunescu. 2006. *Subsequence kernels for relation extraction*. In *Proceedings of NIPS*, pages 171–178.
- Ndapandula Nakashole, Gerhard Weikum, and Fabian Suchanek. 2012. *PATTY: A taxonomy of relational patterns with semantic types*. In *Proceedings of EMNLP-CoNLL*, pages 1135–1145.
- Karthik Narasimhan, Adam Yala, and Regina Barzilay. 2016. *Improving information extraction by acquiring external evidence with reinforcement learning*. In *Proceedings of EMNLP*, pages 2355–2365.
- Dat PT Nguyen, Yutaka Matsuo, and Mitsuru Ishizuka. 2007. *Relation extraction from wikipedia using subtree mining*. In *Proceedings of AAAI*, pages 1414–1420.
- Thien Huu Nguyen and Ralph Grishman. 2015a. *Combining neural networks and log-linear models to improve relation extraction*. *arXiv preprint arXiv:1511.05926*.
- Thien Huu Nguyen and Ralph Grishman. 2015b. *Relation extraction: Perspective from convolutional neural networks*. In *Proceedings of the Workshop of NAACL*, pages 39–48.
- Truc-Vien T Nguyen and Alessandro Moschitti. 2011. *End-to-end relation extraction using distant supervision from external semantic repositories*. In *Proceedings of ACL*, pages 277–282.
- Christina Niklaus, Matthias Cetto, André Freitas, and Siegfried Handschuh. 2018. *A survey on open information extraction*. In *Proceedings of COLING*, pages 3866–3878.
- Ii. I Ntroduction. 2005. *The ace 2005 ( ace 05 ) evaluation plan evaluation of the detection and recognition of ace entities, values, temporal expression, relations, and events*.
- Sachin Pawar, Girish K Palshikar, and Pushpak Bhattacharyya. 2017. *Relation extraction: A survey*. *arXiv preprint arXiv:1712.05191*.
- Nanyun Peng, Hoifung Poon, Chris Quirk, Kristina Toutanova, and Wen-tau Yih. 2017. *Cross-sentence n-ary relation extraction with graph LSTMs*. *TACL*, 5:101–115.
- Jianfeng Qu, Wen Hua, Dantong Ouyang, Xiaofang Zhou, and Ximing Li. 2019. *A fine-grained and noise-aware method for neural relation extraction*. In *Proceedings of CIKM*, pages 659–668.

- Chris Quirk and Hoifung Poon. 2017. Distant supervision for relation extraction beyond the sentence boundary. In *Proceedings of EACL*, pages 1171–1182.
- Sachin Ravi and Hugo Larochelle. 2017. Optimization as a model for few-shot learning. In *Proceedings of ICLR*.
- Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Proceedings of ECML-PKDD*, pages 148–163.
- Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M Marlin. 2013. Relation extraction with matrix factorization and universal schemas. In *Proceedings of NAACL*, pages 74–84.
- Dan Roth and Wen-tau Yih. 2002. Probabilistic reasoning for entity & relation recognition. In *Proceedings of COLING*.
- Dan Roth and Wen-tau Yih. 2004. A linear programming formulation for global inference in natural language tasks. In *Proceedings of CoNLL*.
- Cicero Nogueira dos Santos, Bing Xiang, and Bowen Zhou. 2015. Classifying relations by ranking with convolutional neural networks. In *Proceedings of ACL-IJCNLP*, pages 626–634.
- Sunita Sarawagi and William W Cohen. 2005. Semi-markov conditional random fields for information extraction. In *Proceedings of NIPS*, pages 1185–1192.
- Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *Proceedings of ESWC*, pages 593–607.
- Yusuke Shinyama and Satoshi Sekine. 2006. Preemptive information extraction using unrestricted relation discovery. In *Proceedings of NAACL*, pages 304–311.
- Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Proceedings of NIPS*, pages 4077–4087.
- Richard Socher, Brody Huval, Christopher D Manning, and Andrew Y Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of EMNLP*, pages 1201–1211.
- Stephen Soderland, David Fisher, Jonathan Aseltine, and Wendy Lehnert. 1995. Crystal inducing a conceptual dictionary. In *Proceedings of IJCAI*, pages 1314–1319.
- Linfeng Song, Yue Zhang, et al. 2018. N-ary relation extraction using graph-state lstm. In *Proceedings of EMNLP*.
- Gabriel Stanovsky and Ido Dagan. 2016. Creating a large benchmark for open information extraction. In *Proceedings of EMNLP*, pages 2300–2305.
- Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D Manning. 2012. Multi-instance multi-label learning for relation extraction. In *Proceedings of EMNLP*, pages 455–465.
- Kumutha Swampillai and Mark Stevenson. 2011. Extracting relations within and across sentences. In *Proceedings of RANLP*, pages 25–32.
- Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of ACL*, pages 384–394.
- Shikhar Vashisht, Prince Jain, and Partha Talukdar. 2018. Cesi: Canonicalizing open knowledge bases using embeddings and side information. In *Proceedings of WWW*, pages 1317–1327.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of NIPS*, pages 5998–6008.
- Patrick Verga, David Belanger, Emma Strubell, Benjamin Roth, and Andrew McCallum. 2016. Multilingual relation extraction using compositional universal schema. In *Proceedings of NAACL*, pages 886–896.
- Patrick Verga and Andrew McCallum. 2016. Row-less universal schema. In *Proceedings of ACL*, pages 63–68.
- Patrick Verga, Emma Strubell, and Andrew McCallum. 2018. Simultaneously self-attending to all mentions for full-abstract biological relation extraction. In *Proceedings of NAACL-HLT*, pages 872–884.
- Oriol Vinyals, Charles Blundell, Tim Lillicrap, Daan Wierstra, et al. 2016. Matching networks for one shot learning. In *Proceedings of NIPS*, pages 3630–3638.
- Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Proceedings of CACM*, 57(10):78–85.
- Ngoc Thang Vu, Heike Adel, Pankaj Gupta, et al. 2016. Combining recurrent and convolutional neural networks for relation classification. In *Proceedings of NAACL*, pages 534–539.
- Linlin Wang, Zhu Cao, Gerard De Melo, and Zhiyuan Liu. 2016. Relation classification via multi-level attention cnns. In *Proceedings of ACL*, pages 1298–1307.
- Mengqiu Wang. 2008. A re-examination of dependency path kernels for relation extraction. In *Proceedings of IJCNLP*, pages 841–846.

- Xiaozhi Wang, Xu Han, Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2018. **Adversarial multi-lingual neural relation extraction**. In *Proceedings of COLING*, pages 1156–1166.
- Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. **Knowledge graph embedding by translating on hyperplanes**. In *Proceedings of AAAI*, pages 1112–1119.
- Jason Weston, Antoine Bordes, Oksana Yakhnenko, and Nicolas Usunier. 2013. **Connecting language and knowledge bases with embedding models for relation extraction**. In *Proceedings of EMNLP*, pages 1366–1371.
- Michael Wick, Aron Culotta, et al. 2006. **Learning field compatibilities to extract database records from unstructured text**. In *Proceedings of EMNLP*.
- Ruidong Wu, Yuan Yao, Xu Han, Ruobing Xie, Zhiyuan Liu, Fen Lin, Leyu Lin, and Maosong Sun. 2019. **Open relation extraction: Relational knowledge transfer from supervised data to unsupervised data**. In *Proceedings of EMNLP-IJCNLP*, pages 219–228.
- Shanchan Wu and Yifan He. 2019. **Enriching pre-trained language model with entity information for relation classification**. In *Proceedings of CIKM*, pages 2361–2364.
- Yi Wu, David Bamman, and Stuart Russell. 2017. **Adversarial training for relation extraction**. In *Proceedings of EMNLP*, pages 1778–1783.
- Minguang Xiao and Cong Liu. 2016. **Semantic relation classification via hierarchical recurrent neural network with attention**. In *Proceedings of COLING*, pages 1254–1263.
- Chenyan Xiong, Russell Power, and Jamie Callan. 2017. **Explicit semantic ranking for academic search via knowledge graph embedding**. In *Proceedings of WWW*, pages 1271–1279.
- Kun Xu, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2015a. **Semantic relation classification via convolutional neural networks with simple negative sampling**. In *Proceedings of EMNLP*, pages 536–540.
- Yan Xu, Ran Jia, Lili Mou, Ge Li, Yunchuan Chen, Yangyang Lu, and Zhi Jin. 2016. **Improved relation classification by deep recurrent neural networks with data augmentation**. In *Proceedings of COLING*, pages 1461–1470.
- Yan Xu, Lili Mou, Ge Li, Yunchuan Chen, Hao Peng, and Zhi Jin. 2015b. **Classifying relations via long short term memory networks along shortest dependency paths**. In *Proceedings of EMNLP*, pages 1785–1794.
- Limin Yao, Aria Haghighi, Sebastian Riedel, and Andrew McCallum. 2011. **Structured relation discovery using generative models**. In *Proceedings of EMNLP*, pages 1456–1466.
- Yuan Yao, Deming Ye, Peng Li, Xu Han, Yankai Lin, Zhenghao Liu, Zhiyuan Liu, Lixin Huang, Jie Zhou, and Maosong Sun. 2019. **DocRED: A large-scale document-level relation extraction dataset**. In *Proceedings of ACL*, pages 764–777.
- Zhi-Xiu Ye and Zhen-Hua Ling. 2019. **Multi-level matching and aggregation network for few-shot relation classification**. In *Proceedings of ACL*, pages 2872–2881.
- Katsumasa Yoshikawa, Sebastian Riedel, et al. 2011. **Coreference based event-argument relation extraction on biomedical text**. *Journal of Biomedical Semantics*, 2(S6).
- Xiaofeng Yu and Wai Lam. 2010. **Jointly identifying entities and extracting relations in encyclopedia text via a graphical model approach**. In *Proceedings of ACL*, pages 1399–1407.
- Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. **Kernel methods for relation extraction**. *Proceedings of JMLR*, pages 1083–1106.
- Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. **Distant supervision for relation extraction via piecewise convolutional neural networks**. In *Proceedings of EMNLP*, pages 1753–1762.
- Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. **Relation classification via convolutional deep neural network**. In *Proceedings of COLING*, pages 2335–2344.
- Wenyuan Zeng, Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2017. **Incorporating relation paths in neural relation extraction**. In *Proceedings of EMNLP*, pages 1768–1777.
- Xiangrong Zeng, Shizhu He, Kang Liu, and Jun Zhao. 2018. **Large scaled relation extraction with reinforcement learning**. In *Proceedings of AAAI*, pages 5658–5665.
- Dongxu Zhang and Dong Wang. 2015. **Relation classification via recurrent neural network**. *arXiv preprint arXiv:1508.01006*.
- Min Zhang, Jie Zhang, and Jian Su. 2006a. **Exploring syntactic features for relation extraction using a convolution tree kernel**. In *Proceedings of NAACL*, pages 288–295.
- Min Zhang, Jie Zhang, Jian Su, and Guodong Zhou. 2006b. **A composite kernel to extract relations between entities with both flat and structured features**. In *Proceedings of ACL*, pages 825–832.

Ningyu Zhang, Shumin Deng, Zhanlin Sun, Guanying Wang, Xi Chen, Wei Zhang, and Huajun Chen. 2019a. **Long-tail relation extraction via knowledge graph embeddings and graph convolution networks.** In *Proceedings of NAACL-HLT*, pages 3016–3025.

Shu Zhang, Dequan Zheng, Xinchen Hu, and Ming Yang. 2015. **Bidirectional long short-term memory networks for relation classification.** In *Proceedings of PACLIC*, pages 73–78.

Yuhao Zhang, Peng Qi, and Christopher D. Manning. 2018. **Graph convolution over pruned dependency trees improves relation extraction.** In *Proceedings of EMNLP*, pages 2205–2215.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D Manning. 2017. **Position-aware attention and supervised data improve slot filling.** In *Proceedings of EMNLP*, pages 35–45.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019b. **ERNIE: Enhanced language representation with informative entities.** In *Proceedings of ACL*, pages 1441–1451.

Shubin Zhao and Ralph Grishman. 2005. **Extracting relations with integrated information using kernel methods.** In *Proceedings of ACL*, pages 419–426.

Shun Zheng, Xu Han, Yankai Lin, Peilin Yu, Lu Chen, Ling Huang, Zhiyuan Liu, and Wei Xu. 2019. **DIAG-NRE: A neural pattern diagnosis framework for distantly supervised neural relation extraction.** In *Proceedings of ACL*, pages 1419–1429.

Guodong Zhou, Jian Su, Jie Zhang, and Min Zhang. 2005. **Exploring various knowledge in relation extraction.** In *Proceedings of ACL*, pages 427–434.

Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. **Attention-based bidirectional long short-term memory networks for relation classification.** In *Proceedings of ACL*, pages 207–212.

Hao Zhu, Yankai Lin, Zhiyuan Liu, Jie Fu, Tat-Seng Chua, and Maosong Sun. 2019a. **Graph neural networks with generated parameters for relation extraction.** In *Proceedings of ACL*, pages 1331–1339.

Mengdi Zhu, Zheye Deng, Wenhan Xiong, Mo Yu, Ming Zhang, and William Yang Wang. 2019b. **Towards open-domain named entity recognition via neural correction models.** *arXiv preprint arXiv:1909.06058*.

Zhangdong Zhu, Jindian Su, and Yang Zhou. 2019c. **Improving distantly supervised relation classification with attention and semantic weight.** *IEEE Access*, 7:91160–91168.

# Robustness and Reliability of Gender Bias Assessment in Word Embeddings: The Role of Base Pairs

Haiyang Zhang\*, Alison Sneyd\* and Mark Stevenson

Department of Computer Science

University of Sheffield

haiyang.zhang, a.sneyd, mark.stevenson@sheffield.ac.uk

## Abstract

It has been shown that word embeddings can exhibit gender bias, and various methods have been proposed to quantify this. However, the extent to which the methods are capturing social stereotypes inherited from the data has been debated. Bias is a complex concept and there exist multiple ways to define it. Previous work has leveraged gender word pairs to measure bias and extract biased analogies. We show that the reliance on these gendered pairs has strong limitations: bias measures based off of them are not robust and cannot identify common types of real-world bias, whilst analogies utilising them are unsuitable indicators of bias. In particular, the well-known analogy “man is to computer-programmer as woman is to homemaker” is due to word similarity rather than societal bias. This has important implications for work on measuring bias in embeddings and related work debiasing embeddings.

## 1 Introduction

Word embeddings, distributed representations of words in a low-dimensional vector space, are used in many downstream NLP tasks (Mikolov et al., 2013a,b; Pennington et al., 2014; Peters et al., 2018; Devlin et al., 2019). Recent work has shown they can contain harmful bias and proposed techniques to quantify it (Bolukbasi et al., 2016; Caliskan et al., 2017; Ethayarajh et al., 2019; Gonen and Goldberg, 2019). These techniques leverage cosine similarity to a base pair of gender words, such as (*man, woman*). They include bias measures, which return a magnitude of bias for a given word, and analogies. A well-known example of the latter is “Man is to computer programmer as woman is to homemaker” (Bolukbasi et al., 2016), which has been widely interpreted as demonstrating bias. There have also been related attempts to debias

embeddings (Bolukbasi et al., 2016; Zhao et al., 2018; Dev and Phillips, 2019; Kaneko and Bollegala, 2019; Manzini et al., 2019).

However, to remove bias effectively, an accurate method of identifying it is first required. This is a complex task, not least because the concept of “bias” has multiple interpretations: Mehrabi et al. (2019) identify 23 types of bias that can occur in machine learning applications, including historic (pre-existing in society), algorithmic (introduced by the algorithm) and evaluation (occurs during model evaluation). In the case of word embeddings, it remains an open question if bias identifying techniques reflect social stereotypes in the training data, an artifact of the embedding process or noise. While it is often assumed the first is true, and thus that bias in embeddings can perpetuate harmful stereotypes (Bolukbasi et al., 2016; Caliskan et al., 2017), this has not been conclusively established (Gonen and Goldberg, 2019; Nissim et al., 2019; Ethayarajh et al., 2019). To further complicate matters, multiple methods of quantifying bias have been proposed, often in response to one another’s limitations (see Section 2.1). It is unclear how they compare and which are more reliable.

This work shows that the use of gender base pairs in bias identifying techniques has serious limitations. We propose three criteria to evaluate the performance of gender bias measures using base pairs and systematically compare four popular measures, showing both that they are not robust, and that they do not accurately reflect common types of societal bias. In addition, we demonstrate that the types of analogies proposed in Bolukbasi et al. (2016) are unsuitable indicators of bias; what is ascribed to social bias in analogies is actually an artifact of high cosine similarity in the base pair, which is arguably positive. Our argument is not that embeddings are free of bias; rather it is that bias is a complex problem and current bias measures do

\* denotes equal contribution.

not completely solve it. This has important implications for future work on bias in embeddings and debiasing techniques.

The primary contributions of this work are to: (1) demonstrate the output of gender bias measures is heavily dependant on a chosen gendered base pair (e.g. *(she, he)*) and on the form of a word considered (e.g. singular versus plural); (2) show the measures cannot accurately predict either the socially stereotyped gender of human traits or the correct gender of words when this is encoded linguistically (e.g. lioness); (3) show that analogies generated by gender base pairs (e.g. *(she, he)*) are flawed indicators of bias and the widely-known example “Man is to computer programmer as woman is to homemaker” is not due to gender bias and (4) highlight the complexities of identifying bias in word embeddings, and the limitations of these measures.

## 2 Related Work

### 2.1 Bias Measures

A variety of gender bias measures for word embeddings have been proposed in the literature. Each takes as input a word  $w$  and a gendered base pair (such as *(she, he)*), and returns a numerical output. This output indicates both the magnitude of  $w$ ’s gender bias with respect to the base pair used, and the direction of  $w$ ’s bias (male or female), which is determined by the sign of the score.

**Direct Bias (DB)** (Bolukbasi et al., 2016) defines bias as a projection onto a gender subspace, which is constructed from a set of gender base pairs such as *(she, he)*. The DB of a word  $w$  is computed as  $w_B = \sum_{j=1}^k (\vec{w} \cdot b_j) b_j$ , where  $\vec{w}$  is the embedding vector of  $w$ , the subspace  $B$  is defined by  $k$  orthogonal unit vectors  $b_1, \dots, b_k$  and vectors are normalised. In addition, the authors proposed a method of debiasing embeddings based off of DB.

There is ambiguity in Bolukbasi et al. (2016) about how many base pairs should be used with DB; while experiments to identify bias use only one (namely *(she, he)*), a set of ten is used for debiasing.<sup>1</sup> It is unclear why the particular ten pairs used were chosen, and the extent to which their choice matters. We follow recent work (Gonen and Goldberg, 2019; Ethayarajh et al., 2019) that evaluates DB and focus on the case of a single base

<sup>1</sup>The set of gender-defining pairs used is *{(she, he), (her, his), (woman, man), (mary, john), (herself, himself), (daughter, son), (mother, father), (gal, guy), (girl, boy), (female, male)}*.

pair, i.e.  $k = 1$ . The DB of  $w$  with respect to the gender base pair  $(x, y)$  is then  $\vec{w} \cdot (\vec{x} - \vec{y})$ .

Caliskan et al. (2017) created an association test for word embeddings called WEAT to identify human-like biases. The **Word Association (WA)**, the key component of WEAT, measures the association of  $w$  with two sets of attribute words,  $X$  and  $Y$ . More formally, WA is computed as:

$$\text{mean}_{x \in X} \cos(\vec{w}, \vec{x}) - \text{mean}_{y \in Y} \cos(\vec{w}, \vec{y})$$

To allow for a fair comparison with other methods being evaluated, we focus on the case where the attribute sets contain a single word, i.e.,  $X = \{x\}$  and  $Y = \{y\}$ . Then WA and DB are equivalent as:

$$\cos(\vec{w}, \vec{x}) - \cos(\vec{w}, \vec{y}) = \frac{\vec{w}}{\|\vec{w}\|} \cdot \left( \frac{\vec{x}}{\|\vec{x}\|} - \frac{\vec{y}}{\|\vec{y}\|} \right)$$

Since DB and WA assign a word the same score, we will use **DB/WA** to refer to both measures.

Gonen and Goldberg (2019) argued that bias cannot be directly observed, as assumed in methods such as DB, and that the debiasing method of Bolukbasi et al. (2016) is ineffective. They proposed the **Neighbourhood Bias Metric (NBM)**, which measures the bias of a word  $w$  as the percentage of socially female-biased words and male-biased words among its  $K$  nearest neighbours in a set of predefined gender-neutral words. Setting  $K = 100$ , the NBM bias of a target word  $w$  is measured as:

$$\frac{|\text{female}(w)| - |\text{male}(w)|}{100},$$

where  $\text{female}(w)$  and  $\text{male}(w)$  are sets of socially biased and male words in the neighborhood of  $w$ . The bias direction of words in  $w$ ’s neighborhood is computed using the DB metric with a single base pair. Gonen and Goldberg (2019) use DB with base pair *(she, he)*; our work considers a more general form with base pair  $(x, y)$ .

Ethayarajh et al. (2019) draw attention to the lack of theoretical guarantees surrounding previous work on bias and debiasing. They argue WEAT overestimates bias and is not robust to the choice of defining sets. In addition, and in contrast Gonen and Goldberg (2019), they argue that DB and the debiasing method based off it are effective, but state vectors used with DB should not be normalised. They propose **Relational Inner Product Association (RIPA)** and state that RIPA is most interpretable with a single base pair, a key advantage

of it being that it (unlike WEAT) does not depend on the base pair used. With a single base pair, the RIPA bias of  $w$  with the base pair  $(x, y)$  is:

$$\vec{w} \cdot \left( \frac{\vec{x} - \vec{y}}{\|\vec{x} - \vec{y}\|} \right).$$

## 2.2 Analogies

An alternative approach to identifying gender bias in embeddings is via word analogies. Unlike the gender bias measures discussed in Section 2.1, analogies do not measure the bias of a particular word. Instead, they identify pairs of words which are assumed to have a gendered relationship.

Analogy in word embeddings are important because it has been observed that embedding vectors seem to possess unexpected linear properties: vectors associated with word pairs sharing the same analogical relationship can be identified using vector arithmetic (Mikolov et al., 2013a; Levy and Goldberg, 2014; Ethayarajh et al., 2018). A notable example of this phenomena is  $\overrightarrow{\text{king}} - \overrightarrow{\text{man}} + \overrightarrow{\text{woman}} \approx \overrightarrow{\text{queen}}$  (Mikolov et al., 2013c). This relationship is frequently attributed to a gender difference vector between  $\overrightarrow{\text{man}}$  and  $\overrightarrow{\text{woman}}$ , and between  $\overrightarrow{\text{king}}$  and  $\overrightarrow{\text{queen}}$  (Mikolov et al., 2013c; Ethayarajh et al., 2018). Analogies are considered a benchmark method of measuring the quality of embeddings, though their suitability has been debated (Linzen, 2016; Drozd et al., 2016; Gladkova et al., 2016). The standard approach to solving ‘ $a$  is to  $b$  as  $c$  is to  $?$ ’ is to return:

$$\vec{d}^* = \underset{w \in V'}{\operatorname{argmax}} \operatorname{CosSim}(\vec{w}, \vec{b} - \vec{a} + \vec{c}),$$

where  $V'$  is the embedding vocabulary excluding  $\{a, b, c\}$  (Levy and Goldberg, 2014).

Bolukbasi et al. (2016) proposed using analogies to quantify gender bias in embeddings and proposed a modified analogy task to produce analogies from the gender base pair  $(she, he)$ . The task identifies word pairs  $(x, y)$ , such that “ $he$  is to  $x$  as  $she$  is to  $y$ ”, where  $\|\vec{x} - \vec{y}\| = 1$ . This method was expanded to multi-class forms of bias such as racial bias by Mehrabi et al. (2019). However, the suitability of analogies as indicators of bias was questioned by Nissim et al. (2019), who highlighted the fact that the approach used by Bolukbasi et al. (2016) did not allow analogies to return their input words, thus artificially increasing the perception of bias.

## 3 Approach

Our aim is to examine the extent to which bias identifying techniques are reliably capturing societal gender bias. Bias is a highly complex concept, and although the four bias measures (DB, WA, NBM and RIPA) may detect certain kinds of bias, there is no theoretical guarantee they will detect all forms, that the “bias” they find will be accurate or that different choices of base pair will behave similarly. We therefore explore whether the bias measures are robust in detecting the bias they appear to detect and if there are forms of bias they are not sensitive to. We propose three conditions to test this:

**1) Base pair stability:** If bias measures captured real-world information in a reliable way, it would be expected that reasonable changes of the base pair, such as  $(she, he)$  to  $(woman, man)$  or  $(she, he)$  to  $(She, He)$ , would not frequently cause a significant change in bias.

**2) Word form stability:** While different forms of a word, such as plurals, have different contexts and word vectors, their social bias will not significantly change and they should have similar bias scores.

**3) Linguistic correspondence:** We explore the extent to which the measures predict the expected gender of terms containing explicit gender information (e.g. “lioness”) or, based on some accounts, stereotypically (e.g. “compassionate”).

Of course, due to noise and the problem of implicit bias, these three conditions may not always be true. However, if they do not hold the majority of the time, it must be questioned if the measures are reliably identifying social bias.

## 4 Data

To allow for fair comparisons, we use the same datasets as previous work where possible:

**Embeddings:** 300-dimensional Google News word2vec (Mikolov et al., 2013a,b).

**Professions:** A list of 320 professions (Bolukbasi et al., 2016), often used to analyse bias measures.

**Base pairs:** A standard list of 10 gender base pairs, including  $(she, he)$  (Bolukbasi et al., 2016).

**Gender neutral:** For NBM, we use the set of 26,145 gender neutral words defined in (Gonen and Goldberg, 2019).

In addition, we construct two new test sets, both listed in Appendix A:

**BSRI:** To assess whether word embeddings contain undesirable gender stereotypes, we utilise the Bem Sex Role Inventory (BSRI) which developed

a list of 20 traits for men and 20 for women that are considered to be socially desirable, such as “assertive” and “compassionate” respectively (Bem, 1974).<sup>2</sup> Although derived in the 1970s, this work remains one of the most influential and widely accepted measures of socially constructed gender roles within the social sciences, e.g. (Holt and Ellis, 1998; Dean and Tate, 2016; Starr and Zurbriggen, 2016; Matud et al., 2019). Of particular relevance to NLP applications, Gaucher et al. (2011) use BSRI to identify gender-biased language in job advertisements and demonstrate this language can contribute to workplace gender inequality. BSRI traits not in the embedding vocabulary (e.g. “willing to take risks”) were removed. For each remaining trait, we queried Merriam Webster for other forms of that word (for example, “assertiveness” is a form of “assertive”), resulting in a list of 58 characteristics (27 male and 31 female).

**Animals:** Some words, including the names of certain animals, encode gender linguistically (e.g. “lioness”). Wikipedia provides a table of male and female versions of animal names.<sup>3</sup> This table was downloaded, and duplicates, rare words and terms whose animal usage is uncommon (for example, a “cob” is a male swan) were removed. This resulted a set of 26 terms consisting 13 female-male pairs such as (*hen, rooster*).

## 5 Evaluation

Evaluating gender bias measures is a complex task as there is no inherent ground truth interpretation of the measure’s results. For example, it is unclear when a bias score is problematic. We choose to evaluate the four bias measures (**DB,WA**, **NBM** and **RIPA**) in two ways, first by considering whether a word is assigned a male or female bias, and second what the magnitude of that score is.

The bias direction (male or female) assigned by a measure to a word is determined by the sign of the score (whether a positive score denotes male or female bias depends on the ordering of the base pair words). The assignment of bias direction is viewed an annotation task in which a bias measure (with a specified base pair) is considered an “annotator” making assignments. Consistency between annotators (i.e. versions of bias measures) can be

computed using Cohen’s kappa to determine pairwise agreement (Cohen, 1960) and Fleiss’ kappa (Fleiss, 1971) for multiple annotators. We follow a widely used interpretation of kappa scores (Landis and Koch, 1977).

The second method of evaluation is an analysis of the magnitude of bias assigned. Previous work in this area does not define what constitutes a “significant” change of the magnitude of a bias score. Therefore, we estimate the mean bias in the embedding space as follows: The 50,000 most frequent words in the embedding vocabulary were selected and, following Bolukbasi et al. (2016), all words containing digits, punctuation or that were more than 20 characters long were removed. For each of the remaining 48,088 words, their bias score was calculated with respect to each of the 10 base pairs (so for each measure, there are 480,880 scores). An examination of these scores revealed them to appear approximately normally distributed and so their mean and standard deviation are used as an approximation of the population mean and standard deviation (see Table 1). We consider a relevant change in magnitude to be a change of at least one standard deviation.

	<b>DB/WA</b>	<b>RIPA</b>	<b>NBM</b>
Mean	-0.001	0.024	-0.038
Standard Dev.	0.053	0.239	0.431

Table 1: Mean and standard deviation of bias scores for each measure.

## 6 Results

**Base pair stability:** The first experiment explored the robustness of the four measures (**DB,WA**, **RIPA** and **NBM**) to changing the base pair. For example, Figure 1 illustrates the effects of changing the base pair on the bias score of the word “professor.” More comprehensively, for each bias measure we computed the bias assigned to each profession for each base pair, and then calculated the agreement between the 10 base pairs via Fleiss’ kappa coefficient. The changes in the bias magnitude of a word between base pairs were also computed. Results are shown in Table 2. The level of agreement of bias direction between base pairs was fair (0.29) for NBM and moderate (0.42 and 0.45) for RIPA and DB/WA. This means that changing the base pair frequently caused a profession’s

<sup>2</sup>Our use of BSRI should not be interpreted as an endorsement of these traits as either accurate or desirable; rather we use them as a dataset of commonly held stereotypes.

<sup>3</sup>[https://en.wikipedia.org/wiki/List\\_of\\_animal\\_names](https://en.wikipedia.org/wiki/List_of_animal_names)

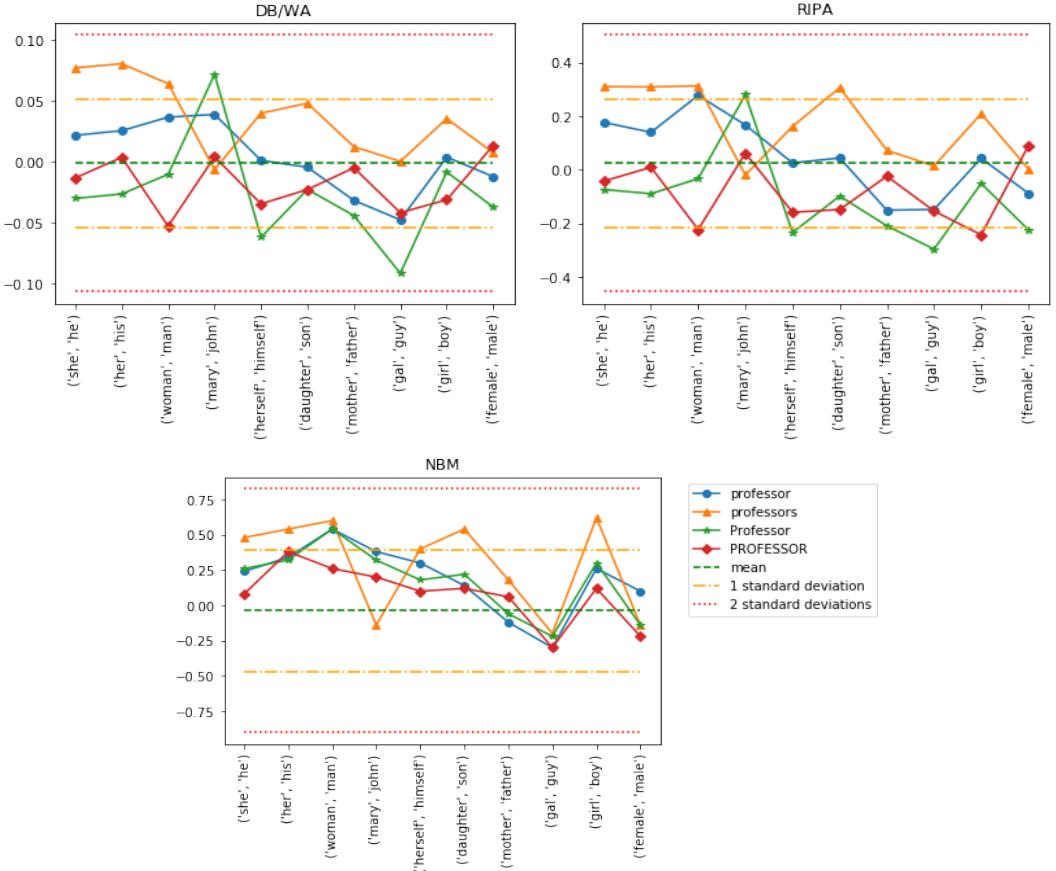


Figure 1: Graphs demonstrating bias score variations. Each graph represents a measure, with the mean and standard deviation of that measure (Section 5) denoted by dashed lines. Positive and negative scores indicate female and male bias respectively, while larger absolute values show higher levels of bias. The bias scores of the word “professor” and its variations (“professors,” Professor” and “PROFESSOR”) are shown, as calculated according to each base pair (such as (*she, he*) and (*her, his*)). The graphs demonstrate that the bias direction and magnitude of bias of each word depend heavily on which base pair is chosen. They also show that the different forms of the word exhibit different behaviour.

	<b>Kappa</b>	<b>Magnitude</b>
DB/WA	<b>0.45</b>	0.69
RIPA	0.42	<b>0.66</b>
NBM	0.29	0.71

Table 2: For the 320 professions 1) the level of agreement kappa between bias directions assigned by each of the ten base pairs and 2) the mean proportion of significant magnitude changes over the 10 base pairs. For 1), higher is better, and for 2), lower is better.

bias direction to change. For example, the RIPA direction of “surgeon” is male for (*she, he*) but female for (*woman, man*). For a given profession, only about a quarter of DB/WA and RIPA directions were the same for every base pair, and fewer than 15% of NBM directions were. With regards to score magnitudes, on average over the professions, 66% of base pair changes saw a relevant change in magnitude (more than one standard deviation) for

RIPA, 69% for DB/WA and 71% for NBM.

Next, to explore the robustness of the form of the base pairs chosen, we compared the bias direction assigned to each of the 320 professions by a base pair to the bias direction assigned by the capitalised form (first letter capitalised) of that base pair (for example, (*she, he*) versus (*She, He*)). The level of agreement of bias direction between each two base pair forms was calculated using Cohen’s kappa coefficient, results are shown in Table 3. The mean of the level of agreement over each of the 10 base pairs ranged from 0.39 (fair) to 0.43 (moderate), with many individual agreements below moderate level. In particular, an agreement level of only 0.03 (very slight) is found for the base pair (*gal, guy*) compared with (*Gal, Guy*) for DB/WA.

**Word form stability:** The second experiment examined the measures’ robustness to changing the form of a word considered by comparing a word’s plural, capitalised (first letter capitalised) and up-

percase (all letters capitalised) forms to its base form. For example, “professors,” “Professor” and “PROFESSOR” were compared to “professor” (see Figure 1). For this experiment, only the 230 words in the professions list whose plural, capitalised and uppercase forms are all included in the embedding vocabulary were used.

For each measure and base pair, the direction of gender bias of each word form was computed, and the pairwise level of agreement (Cohen’s kappa) between the original form of a word and each of its variants was calculated, see Table 4. All four measures were found to give different versions of the same word (plural, capital and uppercase forms) different bias directions. For example, the DB/WA of “surgeon” is male but of “surgeons” is female (base pair (*she, he*)). For each measure, the mean kappa coefficients were moderate for the plural category and fair for the uppercase category. For the capital category, they were moderate for DB/WA and RIPA, and substantial for NBM. Since changing word form frequently changes bias direction, these results indicate the bias measures are not reliably reflecting any inherent social bias encoded into the word vectors, and that the gender bias direction assigned to a profession is not robust.

**Linguistic Correspondence:** The final experiment examined the measures’ prediction for terms containing explicit or stereotypical gender information, in the form of social stereotypes (BSRI) and linguistic gender (Animals). The predicted gender bias direction of the words in the Animals and BSRI lists was computed for each base pair and measure, and compared with the ground-truth gender of the words. Table 5 shows the pairwise agreement (Cohen’s kappa) between prediction and ground-truth for each base pair, as well as the mean agreement over all 10 base pairs.

The bias measures did not predict the ground-truth gender of either set of words with high accuracy; mean agreement levels varied from 0.17 (slight) to 0.42 (moderate). For example, the NBM gender prediction for “bull,” a male animal, was female and the direction of the feminine BSRI trait “compassionate” was male (both for base pair (*woman, man*)). As with the previous experiment, different forms of the BSRI words frequently were assigned opposite genders: unlike “compassionate”, “compassionately” had the correct NBM gender prediction, again with base pair (*woman, man*). The BRSI results were overall

poorer than the Animal results, with some base pairs having negative kappa scores, indicating less agreement than random chance. This may be because the BSRI stereotypes are less likely to be mentioned in the context of base pair words like “he” and “she.” Interestingly, the highest scoring BSRI base pair was (*mother, father*). Some of the inaccurate predictions for the animal words may come from the fact that some terms can both refer to males and be gender neutral, e.g. “lion.”

## 7 Discussion

**Lack of Robustness:** The experiments in this work empirically showed that the four bias measures are not robust to changing either the base pair or the form of a word used (such as singular to plural). We hypothesise there are two primary reasons for this: sociolinguistic factors and mathematical properties of the bias measure formulae.

It is highly likely that linguistic properties of the base pair chosen effect bias measure robustness.<sup>4</sup> For example, (*she, he*) has quite different sociolinguistic connotations to the more casual (*gal, guy*), and “she” and “he” are clearly linguistic opposites, unlike “Mary” and “John.” Our results indicate that more neutral base pairs which are linguistic opposites, such as (*she, he*) or (*man, woman*) are the most robust. However, even they exhibit variation and struggle particularly to pick up on social stereotypes (the BSRI agreements for (*man, woman*) are all close to zero, indicating random chance).

A further reason that the bias measures are not robust is their reliance on the direct output of a dot product, which is sensitive to the input vectors used. Given a base pair (*a, b*), we will refer to  $\vec{a} - \vec{b}$  as its difference vector. The 10 base pairs have highly similar difference vectors: the mean over the 10 base pairs of  $\cos(\vec{a} - \vec{b}, \vec{c} - \vec{d})$ , where (*a, b*) and (*c, d*) are base pairs is 0.5. While this is very high for embedding vectors,<sup>5</sup> it does not guarantee  $\vec{w} \cdot (\vec{x} - \vec{y})$  and  $\vec{w} \cdot (\vec{a} - \vec{b})$  will have the same sign for all words *w*, resulting in opposite bias directions. The same sensitivity explains why words and their plurals can be assigned opposite bias directions, even if they have similar embeddings. Furthermore, similarity between base pair difference vectors is highly correlated with agree-

---

<sup>4</sup>Our choice of base pairs follows previous work.

<sup>5</sup>We randomly sampled 100,000 sets of words  $\{a, d, c, d\}$  and computed  $\cos(\vec{a} - \vec{b}, \vec{c} - \vec{d})$ ; the sample mean was 0.00, with standard deviation 0.09.

	She	Her	Woman	Mary	Herself	Dgtr	Mother	Gal	Girl	Female	Mean
	He	His	Man	John	Himself	Son	Father	Guy	Boy	Male	
DB/WA	<b>0.65</b>	0.53	0.56	0.32	0.60	0.28	0.40	0.03	0.49	0.38	0.42
RIPA	<b>0.80</b>	0.56	0.58	0.32	0.59	0.27	0.31	0.04	0.49	0.35	0.43
NBM	0.58	0.65	0.61	0.19	<b>0.69</b>	0.18	0.23	0.10	0.53	0.18	0.39

Table 3: Results of the base pair stability experiments: Agreement between the bias directions assigned by a base pair and its capitalised form (e.g. (she,he) and (She, He)) for the 320 professions, and the mean over all base pairs.

	she	her	woman	mary	herself	dgtr	mother	gal	girl	female	Mean	
	he	his	man	john	himself	son	father	guy	boy	male		
Plural	DB/WA	0.50	0.51	<b>0.53</b>	0.35	0.47	0.33	0.42	0.47	0.52	<b>0.53</b>	0.46
	RIPA	0.57	0.58	<b>0.63</b>	0.39	0.53	0.46	0.44	0.53	0.53	0.50	0.52
	NBM	0.69	0.57	<b>0.72</b>	0.38	0.65	0.32	0.50	0.59	0.60	0.62	0.56
Capital	DB/WA	0.61	0.66	0.59	0.42	0.67	<b>0.79</b>	0.61	0.50	0.50	0.44	0.58
	RIPA	0.60	0.60	0.54	0.36	0.59	<b>0.69</b>	0.61	0.54	0.53	0.45	0.55
	NBM	<b>0.77</b>	0.63	0.68	0.54	0.74	0.68	0.61	0.71	0.65	0.63	0.66
Upper	DB/WA	0.19	0.35	0.43	0.17	0.29	<b>0.48</b>	0.18	0.20	0.34	0.30	0.29
	RIPA	0.35	0.38	0.40	0.16	0.35	<b>0.53</b>	0.22	0.20	0.30	0.27	0.32
	NBM	0.50	0.52	0.49	0.25	0.52	0.40	0.22	0.46	<b>0.54</b>	0.13	0.40

Table 4: Results of the word form stability experiments: Agreement between the bias direction of a profession and its plural, capital and uppercase forms for each base pair, and the mean over all base pairs.

	she	her	woman	mary	herself	dgtr	mother	gal	girl	female	Mean	
	he	his	man	john	himself	son	father	guy	boy	male		
BSRI	DB/WA	0.35	0.37	0.07	-0.03	0.14	0.03	<b>0.45</b>	0.39	-0.08	0.01	0.17
	RIPA	0.44	0.40	0.09	-0.08	0.12	0.16	<b>0.45</b>	0.39	-0.08	0.01	0.19
	NBM	0.27	0.32	-0.01	0.01	0.27	0.17	<b>0.46</b>	0.14	0.18	-0.04	0.18
Animal	DB/WA	<b>0.54</b>	0.38	<b>0.54</b>	<b>0.54</b>	<b>0.54</b>	0.31	0.23	0.46	<b>0.54</b>	0.08	0.42
	RIPA	0.31	0.38	0.31	0.46	0.46	0.23	0.23	<b>0.54</b>	0.46	0.08	0.35
	NBM	0.31	0.08	0.15	0.15	0.15	0.00	0.08	<b>0.46</b>	0.15	0.00	0.15

Table 5: Results of the linguistic correspondence experiments: Agreement between the ground-truth and predicted gender for each base pair, and the mean over all 10 base pairs.

	she	her	woman	mary	herself	dgtr	mother	gal	girl	female	Mean
	he	his	man	john	himself	son	father	guy	boy	male	
DB/WA & RIPA	0.69	0.86	0.64	0.90	0.82	0.79	0.92	0.85	0.89	<b>0.96</b>	0.83
DB/WA & NBM	0.54	0.37	<b>0.62</b>	0.44	0.55	0.54	0.46	0.34	0.48	0.47	0.48
RIPA & NBM	0.52	0.42	<b>0.66</b>	0.41	0.57	0.57	0.47	0.29	0.47	0.50	0.49

Table 6: Comparing bias measures: Agreement between the bias direction assigned by each pair of bias measures (with a fixed base pair) for the 320 professions, and the mean over the 10 base pairs.

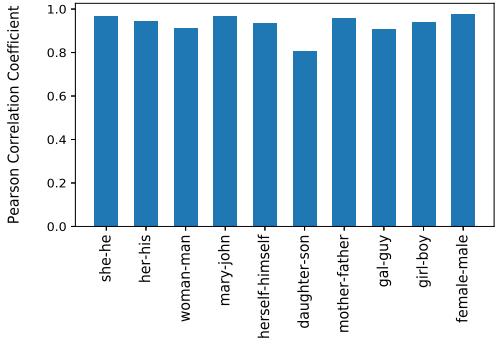


Figure 2: Correlation between the cosine similarity of the base pair difference vectors and the corresponding pairwise kappa coefficients for the DB/WA professions bias directions.

ment between bias directions: For each base pair  $(a, b)$ , we computed  $\cos(\vec{a} - \vec{b}, \vec{c} - \vec{d})$ , for each of the other 9 base pairs  $(c, d)$ , and compared these scores to the pairwise agreements between the corresponding DB/WA bias directions assigned to the professions. There was a high Pearson correlation (max p-value 0.005) in each case, see Figure 2.

The lack of robustness of the gender bias measures means care should be taken in ascribing their output to historic bias in the training data or algorithmic bias in the embedding process. Rather, our analysis indicates that a significant proportion of the “bias” found is an artifact of the evaluation method (bias measures) used.

**Comparing Bias Measures:** A limitation of previous work is it unclear which of the proposed gender bias measures is best, even though they are often introduced as alternatives to one another. The results of our study are mixed and no one measure emerges as reliable.

Despite NBM being designed as an alternative to DB, which takes into account the socially biased neighbours of a word, our experiments found it performs more poorly on the socially biased terms (BSRI) than DB with its recommended base pair (*she, he*) (Table 5). Conversely, it was less sensitive to different word-forms (Table 4). This is likely because different forms of  $w$  share common subsets of top  $K$ -neighbors with  $w$ . Furthermore, Ethayarajh et al. (2019) claim RIPA is an improvement on WA because RIPA is robust to changing the base pair if the two corresponding difference vectors are “roughly the same,” and give (*man, woman*) and (*king, queen*) as an example. However, we find this claim does not hold: this change of base pair causes 28% (91) of the Professions words to alter their RIPA bias direction.

Finally, we compared agreement between the bias directions assigned to the professions by different pairs of measures (Table 6). The results show that on average, there is an almost perfect level of agreement (0.83) between RIPA and DB/WA, and moderate levels of agreement between NBM and the other measures. As RIPA and DB/WA have very similar formulae, the high level of agreement between them for each base pair indicates that the choice of base pair is highly influential and more important than the difference in their formulae. Figure 1 illustrates this point by showing that the measures tend to change in a similar manner from base pair to base pair for each word variant.

**Analogies do not indicate bias:** Analogies are often used as evidence of bias in word embeddings (Bolukbasi et al., 2016; Manzini et al., 2019). This section argues they are unsuitable indicators of bias as they primarily reflect similarity, and not necessarily linguistic relationships like gender. More formally, given an analogy “ $a$  is to  $b$  as  $c$  is to  $?$ ,” we show, using *multi-dimensional vector-valued functions* (Larson and Edwards, 2016), that if there is a high cosine similarity between  $a$  and  $c$ , the predicted answer will be a word similar to  $b$ .

Suppose a function  $F : \mathbb{R}^m \rightarrow \mathbb{R}^n$  has component functions  $f_i : \mathbb{R}^m \rightarrow \mathbb{R}$ ,  $i \in \{1, \dots, n\}$ , where  $F(\vec{x}) = (f_i(\vec{x}))_{i=1}^n$  and  $\vec{x} = (x_j)_{j=1}^m$ . Then the limit of  $F$ , if it exists, can be found by taking the limit of each component function:

$$\lim_{\vec{x} \rightarrow \vec{a}} F(\vec{x}) = \left( \lim_{\vec{x} \rightarrow \vec{a}} f_i(\vec{x}) \right)_{i=1}^n.$$

For fixed vectors  $\vec{a}, \vec{b} \in \mathbb{R}^n$ , let  $F : \mathbb{R}^n \rightarrow \mathbb{R}^n$ ,  $\vec{x} \mapsto \vec{x} - \vec{a} + \vec{b}$ .  $F$  can be expressed component-wise as  $F(\vec{x}) = (f_i(\vec{x}))_{i=1}^n = (x_i - a_i + b_i)_{i=1}^n$ . Then as each component function is continuous:

$$\begin{aligned} \lim_{\vec{x} \rightarrow \vec{a}} F(\vec{x}) &= \left( \lim_{\vec{x} \rightarrow \vec{a}} (x_i - a_i + b_i) \right)_{i=1}^n \\ &= (a_i - a_i + b_i)_{i=1}^n \\ &= \vec{b}. \end{aligned}$$

Thus as  $\vec{x}$  approaches  $\vec{a}$ ,  $\vec{x} - \vec{a} + \vec{b}$  approaches  $\vec{b}$ . For embeddings, this means if  $\vec{a}$  is sufficiently similar to  $\vec{c}$ , by Equation 2.2, we expect the predicted answer  $d^*$  to the analogy “ $a$  is to  $b$  as  $c$  is to  $?$ ” to be a word whose vector is similar to  $\vec{b}$ . This was demonstrated empirically in (Linzen, 2016).

Implications of the well-known analogy “*man* is to *computer programmer* as *woman* is to

*homemaker*” should be reinterpreted in light of this insight. Previous interpretations took this analogy to be evidence of systematic gender bias in the embedding space (Bolukbasi et al., 2016). However, there is a very high cosine similarity between  $\overrightarrow{\text{man}}$  and  $\overrightarrow{\text{woman}}$  (0.77)<sup>6</sup>; in fact, each is the most similar word to the other in the embedding space. The vectors for *computer programmer* and *homemaker* are also highly similar (0.50). The presence of *homemaker* can therefore be explained by its similarity to *computer programmer* rather than gender bias. Of course, embedding vector similarity does frequently indicate word relatedness (e.g. “king” and “queen”). However, vector similarity may also be due to noise. As there is no obvious linguistic relationship between the words *homemaker* and *computer programmer* and neither are common words in the embedding vocabulary, we posit the latter is the case.

This analogy has been taken as evidence of a gendered relationship between *computer programmer* and *homemaker* because it has been assumed that the principal relation between the vectors for *man* and *woman* is gender, and that this relation carries over to *computer programmer* and *homemaker*. This argument rests on the supposition that the difference vector  $\overrightarrow{\text{man}} - \overrightarrow{\text{woman}}$  encodes gender. However, embeddings were not designed to have such linear properties and their existence has been debated (Linzen, 2016). Furthermore, the top solution for “*man* is to *apple* as *woman* is to ?” is *apples*, but the relationship between *apple* and *apples* is clearly pluralisation rather than gender. More generally, we took the commonly used Google Analogy Test Set (Mikolov et al., 2013a) which contains 19,544 analogies (8,869 semantic and 10,675 syntactic) split into 14 categories, such as countries and their capitals. This set contains 550 unique word pairs  $(x, y)$  (such as  $(\text{apple}, \text{apples})$ ) unrelated to gender.<sup>7</sup> In general, the two words in each of the 550 word pairs are highly similar to each other, with mean cosine similarity 0.62 and standard deviation 0.13. We tested the analogy “*man* is to *x* as *woman* is to ?” using Equation 2.2. This resulted

<sup>6</sup>By comparison, the mean cosine similarity for 100,000 pairs of words randomly sampled from the embedding space was 0.13, with standard deviation 0.11.

<sup>7</sup>The category “family” was excluded as there are gender relationships between the word pairs.

in 22% being correctly solved (i.e. returning *y*), including 76% correct in the “gram8-plural” category, which contains pluralised words (note that the analogy not being solved correctly does not imply a dissimilar vector is being returned). This demonstrates that “*man* is to *x* as *woman* is to ?” frequently solves analogies by returning words whose vectors are similar to  $\overrightarrow{x}$ , without any need for a linguistically gendered relationship between *x* and the returned word.

These observations have further implications for the biased analogy generating method of Bolukbasi et al. (2016), which was extended in (Manzini et al., 2019). This method leveraged the base pair  $(\text{she}, \text{he})$  to find word pairs  $(x, y)$ , such that “*he* is to *x* as *she* is to *y*”, where  $\|\overrightarrow{x} - \overrightarrow{y}\| = 1$ . However, the condition  $\|\overrightarrow{x} - \overrightarrow{y}\| = 1$  is equivalent to  $\cos(\overrightarrow{x}, \overrightarrow{y}) = \frac{1}{2}$ . This forced similarity between *x* and *y* combined with the high similarity of *she* and *he* (0.61) means this method is simply returning word pairs with a high similarity. Alternative choices of gender base pair such as  $(\text{woman}, \text{man})$  would suffer from the same flaw. Consequently, analogies produced using this method should be treated with caution.

## 8 Conclusions

There has been a recent focus in the NLP community on identifying bias in word embeddings. While we strongly support the aim of such work, this paper highlights the complexity of trying to quantify bias in embeddings. We showed the reliance of popular gender bias measures on gender base pairs has strong limitations. None of the measures are robust enough to reliably capture social bias in embeddings, or to be leveraged in debiasing methods. In addition, we showed the use of gender base pairs to generate “biased” analogies is flawed. Our analysis can contribute to future work designing robust bias measures and effective debiasing methods. Although this paper focused on gender bias, it is relevant to work examining other forms of bias, such as racial stereotyping, in embeddings. Code to replicate our experiments can be found at: [https://github.com/alisonsneyd/Gender\\_bias\\_word\\_embeddings](https://github.com/alisonsneyd/Gender_bias_word_embeddings)

## Acknowledgements

This work was supported by the Institute of Coding which received funding from the Office for Students (OfS) in the United Kingdom.

## References

- Sandra L. Bem. 1974. **The measurement of psychological androgyny.** *Journal of consulting and clinical psychology*, 42:155–62.
- Tolga Bolukbasi, Kai-Wei Chang, James Zou, Venkatesh Saligrama, and Adam Kalai. 2016. **Man is to computer programmer as woman is to homemaker? debiasing word embeddings.** In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS’16, pages 4356–4364, USA. Curran Associates Inc.
- Aylin Caliskan, Joanna J Bryson, and Arvind Narayanan. 2017. **Semantics derived automatically from language corpora contain human-like biases.** *Science*, 356(6334):183–186.
- Jacob Cohen. 1960. Coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37—46.
- M. Dean and Charlotte Tate. 2016. **Extending the legacy of sandra bem: Psychological androgyny as a touchstone conceptual advance for the study of gender in psychological science.** *Sex Roles*, 76.
- Sunipa Dev and Jeff M. Phillips. 2019. **Attenuating bias in word vectors.** In *The 22nd International Conference on Artificial Intelligence and Statistics, AISTATS 2019, 16-18 April 2019, Naha, Okinawa, Japan*, pages 879–887.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. **BERT: Pre-training of deep bidirectional transformers for language understanding.** In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- Aleksandr Drozd, Anna Gladkova, and Satoshi Mat-suoka. 2016. **Word embeddings, analogies, and machine learning: Beyond king - man + woman = queen.** In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3519–3530, Osaka, Japan. The COLING 2016 Organizing Committee.
- Kawin Ethayarajh, David Duvenaud, and Graeme Hirst. 2018. **Towards understanding linear word analogies.** *CoRR*, abs/1810.04882.
- Kawin Ethayarajh, David Duvenaud, and Graeme Hirst. 2019. Understanding undesirable word embedding associations. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 1696–1705. Association for Computational Linguistics.
- Joseph L. Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382.
- Danielle Gaucher, Justin P Friesen, and Aaron C. Kay. 2011. Evidence that gendered wording in job advertisements exists and sustains gender inequality. *Journal of personality and social psychology*, 101:109–28.
- Anna Gladkova, Aleksandr Drozd, and Satoshi Mat-suoka. 2016. **Analogy-based detection of morphological and semantic relations with word embeddings: what works and what doesn’t.** In *Proceedings of the NAACL Student Research Workshop*, pages 8–15, San Diego, California. Association for Computational Linguistics.
- Hila Gonen and Yoav Goldberg. 2019. **Lipstick on a pig: Debiasing methods cover up systematic gender biases in word embeddings but do not remove them.** In *NAACL-HLT*.
- Cheryl L Holt and Jon B Ellis. 1998. Assessing the current validity of the bem sex-role inventory. *Sex roles*, 39(11-12):929–941.
- Masahiro Kaneko and Danushka Bollegala. 2019. **Gender-preserving debiasing for pre-trained word embeddings.** In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1641–1650, Florence, Italy. Association for Computational Linguistics.
- J. Richard Landis and Gary G. Koch. 1977. **The measurement of observer agreement for categorical data.** *Biometrics*, 33(1):159–174.
- R. Larson and B.H. Edwards. 2016. *Calculus*. Cengage Learning.
- Omer Levy and Yoav Goldberg. 2014. **Linguistic regularities in sparse and explicit word representations.** In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pages 171–180, Ann Arbor, Michigan. Association for Computational Linguistics.
- Tal Linzen. 2016. **Issues in evaluating semantic spaces using word analogies.** In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*, pages 13–18, Berlin, Germany. Association for Computational Linguistics.
- Thomas Manzini, Yao Chong Lim, Yulia Tsvetkov, and Alan W. Black. 2019. **Black is to criminal as caucasian is to police: Detecting and removing multi-class bias in word embeddings.** In *NAACL-HLT*.
- M. Pilar Matud, Marisela López-Curbelo, and Demelza Fortes. 2019. **Gender and psychological well-being.** *International Journal of Environmental Research and Public Health*, 16(19):3531.
- Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. 2019. **A survey on bias and fairness in machine learning.**

Tomas Mikolov, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. *Distributed representations of words and phrases and their compositionality*. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS’13, pages 3111–3119, USA. Curran Associates Inc.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013c. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751.

Malvina Nissim, Rik van Noord, and Rob van der Goot. 2019. Fair is better than sensational: Man is to doctor as woman is to doctor. *CoRR*, abs/1905.09866.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. *Glove: Global vectors for word representation*. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. *Deep contextualized word representations*. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Christine Starr and Eileen Zurbriggen. 2016. Sandra bem’s gender schema theory after 34 years: A review of its reach and impact. *Sex Roles*.

Jieyu Zhao, Yichao Zhou, Zeyu Li, Wei Wang, and Kai-Wei Chang. 2018. *Learning gender-neutral word embeddings*. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4847–4853, Brussels, Belgium. Association for Computational Linguistics.

## A Appendix

**BSRI Female Terms:** affectionate, affectionately, cheerful, cheerfully, cheerfulness, childlike, compassionate, compassionately, feminine, femininely, gentle, gently, gullible, gullibility, gullibly, loyal, loyally, shy, shyly, shyness, sympathetic, sympathetically, tender, tenderly, tenderness, understanding, understandingly, warm, warmish, warmth, yielding

**BSRI Male Terms:** aggressive, aggressively, aggressiveness, aggressivity, ambitious, ambitiously,

ambitiousness, analytical, analytically, assertive, assertiveness, assertively, athletic, athleticism, athletically, competitive, competitiveness, competitively, dominant, dominantly, forceful, forcefulness, independent, independently, individualistic, masculine, selfsufficient

**Female Animal Terms:** bitch, cow, doe, duck, ewe, goose, hen, leopardess, lioness, mare, queen, sow, tigress

**Male Animal Terms:** dog, bull, buck, drake, ram, gander, rooster, leopard, lion, stallion, drone, boar, tiger

# ExpanRL: Hierarchical Reinforcement Learning for Course Concept Expansion in MOOCs

Jifan Yu<sup>1</sup>, Chenyu Wang<sup>1</sup>, Gan Luo<sup>1</sup>,  
Lei Hou<sup>1,2,3\*</sup>, Juanzi Li<sup>1,2,3</sup>, Jie Tang<sup>1,2,3</sup>, Minlie Huang<sup>1,2,3</sup>, Zhiyuan Liu<sup>1,2,3</sup>

<sup>1</sup>Dept. of Computer SCi.& Tech., Tsinghua University, China 100084

<sup>2</sup>KIRC, Institute for Artificial Intelligence, Tsinghua University, China 100084

<sup>3</sup>Beijing National Research Center for Information Science and Technology, China 100084

{yujf18, luog18}@mails.tsinghua.edu.cn  
{houlei, lijuanzi, jietang, aihuang, liuzy}@tsinghua.edu.cn

## Abstract

Within the prosperity of Massive Open Online Courses (MOOCs), the education applications that automatically provide extracurricular knowledge for MOOC users have become rising research topics. However, MOOC courses' diversity and rapid updates make it more challenging to find suitable new knowledge for students. In this paper, we present ExpanRL, an end-to-end hierarchical reinforcement learning (HRL) model for concept expansion in MOOCs. Employing a two-level HRL mechanism of seed selection and concept expansion, ExpanRL is more feasible to adjust the expansion strategy to find new concepts based on the students' feedback on expansion results. Our experiments on nine novel datasets from real MOOCs show that ExpanRL achieves significant improvements over existing methods and maintain competitive performance under different settings.

## 1 Introduction

The cognitive-driven theory has been widely used in practical teaching since Ausubel firstly proposed it in (Ausubel, 1968), which suggests educators provide new knowledge for students to motivate their learning continuously. In fact, in addition to the concepts taught in course, many related concepts are also attractive and worthy of learning. As shown in Figure 1, when a student studies the concept *LSTM* in “Deep Learning” course from Coursera<sup>1</sup>, many related concepts, including its prerequisite concepts (*RNN*), related scientists (*Jürgen Schmidhuber*) and its related applications (*Machine Translation*) can also benefit his/her further study. In traditional classrooms, these concepts are often considerably introduced by teachers.

\*Corresponding author.

<sup>1</sup><https://www.coursera.org>

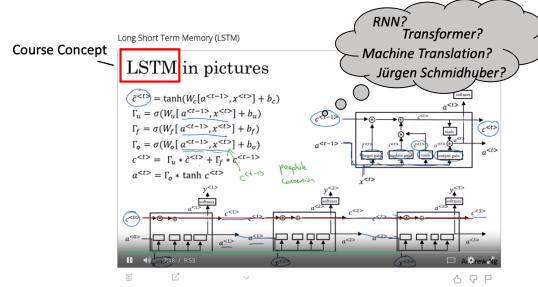


Figure 1: An example of course-related concepts in the “Deep Learning” course from Coursera.

However, in the era of Massive Open Online Courses (MOOCs), thousands of courses are pre-recorded for with millions of students with various backgrounds (Shah, 2019), which makes it infeasible to pick out these essential concepts manually. Therefore, there is a clear need to automatically discover course-related concepts so that they can easily acquire additional knowledge and achieve better educational outcomes.

This task is formally defined as **Course Concept Expansion** (Yu et al., 2019a), a special type of *Concept Expansion* or *Set Expansion* (Wang and Cohen, 2007), which refers to the task of expanding a small set of seed concepts into a complete set of concepts that belong to the same course or subject from external resources. Despite abundant efforts in related topics (He and Xin, 2011; Shen et al., 2017; Yan et al., 2019), existing methods still face three challenges when applied to MOOCs.

First, distinct from the task of enriching a certain concept set, the purpose of course concept expansion is to benefit students' learning, making the context information insufficient to detect whether a concept is appropriate to be an expansion result. How to properly introduce student feedback in the model's loop is a crucial challenge.

Second, unlike the set expansion for a clear general category (e.g., countries), courses are often the

combinations of multiple categories, especially in interdisciplinary courses like *Mathematics for Computer Science*<sup>2</sup>. Therefore, it isn't easy to model the course's semantic scope (Curran et al., 2007) when applying existing expansion methods.

Third, MOOCs are updated continuously, and numerous new courses arise everyday (Shah, 2019), which requires a good generalization ability of the expansion model; otherwise, the frequent model retraining will cause severe waste of resources.

To address the above problems, we construct a novel interactive environment on real MOOCs, which collects students' feedback on expansion results and provides new knowledge for MOOC students in an interesting way for better education. And based on the feedback, we propose ExpanRL, a hierarchical reinforcement learning framework for course concept expansion in MOOCs, which decomposes the concept expansion task into a hierarchy of two subtasks: high-level seed selection and low-level expansion.

Boosted by user feedback on expansion results, ExpanRL jointly learns how to select seed concepts to model the semantic scope of the course better, and whether a concept is beneficial for students. Moreover, the hierarchical reinforcement learning (HRL) structure enables ExpanRL to learn proper expansion strategies instead of the modeling of a particular course, making our model keep a high performance even in unobserved courses.

The evaluation is conducted on 9 datasets from real MOOC courses, compared with 5 representative baseline methods. We further conduct an online evaluation to investigate whether students admit the expanded concepts.

Our contributions include 1) an investigation on how to involve HRL framework into the task of concept expansion; 2) a paradigm that connects the NLP concept expansion task with the educational application; 3) an interactive MOOC environment, consisting of 9 novel datasets of different subjects, 6,553 extracted course concepts, and 495,324 user behaviors from a real MOOC website.

## 2 Preliminaries

### 2.1 Problem Formulation

Following (Yu et al., 2019a), **Course Concept Expansion** is formally defined as: given the course corpus  $\mathcal{D}$ , course concepts  $\mathcal{M}$ , and a knowledge

base  $\mathcal{KB}$  as an external source, the task is to return a ranked list of expanded concepts  $E_c$ .

In this formulation, a course corpus is defined as  $\mathcal{D} = \{\mathcal{C}_j\}_{j=1}^{|\mathcal{N}|}$ , which is composed of  $n$  courses' video subtitles in the same subject area. Course concepts are the subjects taught in the course (such as *LSTM* in Figure 1), denoted as  $\mathcal{M} = \{c_i\}_{i=1}^{|\mathcal{M}|}$ . (Pan et al., 2017). Knowledge base  $\mathcal{KB} = (E, R)$  is consist of concepts  $E$  and relations  $R$ , which is utilized as an external source to obtain expansion candidates. Though other source (such as Web tables) can also take on this role, we still employ a  $\mathcal{KB}$  to search for expansion candidates like the prior work, i.e.,  $E_c \subset E$ .

### 2.2 Basic Model for Concept Expansion

The general idea of concept expansion is first to characterize the concept set according to its representative elements, then find new candidates and rank them to expand the set.

**Seed Selection Stage.** A group of representative concepts are called *seeds* and formalized to  $K \subset E_c$  (Wang and Cohen, 2007; Mamou et al., 2018). While the expansion process is often carried out iteratively, we also formalize the expansion set of round  $t$  to  $E_c^t$ . Seed selection is to calculate the possibility that each concept in  $E_c^t$  becomes a seed, i.e.,  $P(c_i \in K^t \subset E_c^t | t)$ , where  $K^t$  contains the seeds of  $t$ -th round.

Based on these *seeds*, we can extract features of the current set and search for candidate concepts for expansion from external sources.

**Expansion Stage.** After finding a new list of candidates  $\mathcal{L}^t = \{c_1, \dots, c_{t'}, \dots, c_{|\mathcal{L}^t|}\}$ , expansion stage aims to calculate the likelihood of  $c_{t'}$  to be a expanded concept. The top candidates ranked by  $c_{t'}$  are selected as new expanded concepts, denoted as  $N^t$  the likelihood can be formalized as  $P(c_{t'} \in N^t \subset \mathcal{L}^t | K^t, t')$ .

The expansion set is refreshed as  $E_c^{t+1} = E_c^t \cup N^t$  until its size reaches the preset upper limit  $\tau$  or cannot find new candidates (He and Xin, 2011).

### 2.3 Interactive MOOC Environment

The workflow above has been experimentally proven to be effective in many concept expansion tasks (Shen et al., 2018; Rastogi et al., 2019). However, such methods only consider the course concepts' semantic information, which makes their expansion results hard to match real learning needs, especially when dealing with the multi-category

<sup>2</sup>A course from the University of London in Coursera.

MOOC courses. Meanwhile, since the models are trained before launching, how to maintain high performance on new arisen courses is challenging. Yu et al. (2019a) designs an online game in MOOCs to collect user feedback on the expansion result, thereby employing an active pipeline model to face the above problems, which provides an interactive MOOC environment for reinforcement learning models.

However, the size of publicly published datasets (4 courses with 800 concepts in each course) is still insufficient to meet the need to train advanced deep learning models. Therefore, we extract 68 real MOOC courses of six subjects and build a large-scale MOOC interactive environment, which contains a gamefied interface for feedback collection and several course datasets: “Mathematics”, “Chemistry”, “Architecture”, “Psychology”, “Material Science” and “Computer Science”, covering diverse subjects of natural science, social science and engineering. The details of the datasets are presented in the experiment section.

We construct the environment through three stages. First, for each subject, we select its most relevant courses from a real MOOC website<sup>3</sup>. We use the method of Pan (2017) to extract the course concepts and manually select the high-quality ones as the course concepts  $\mathcal{M}$ . Second, we take XLORE (Jin et al., 2019) as  $\mathcal{KB}$  to search for candidate expansion concepts.

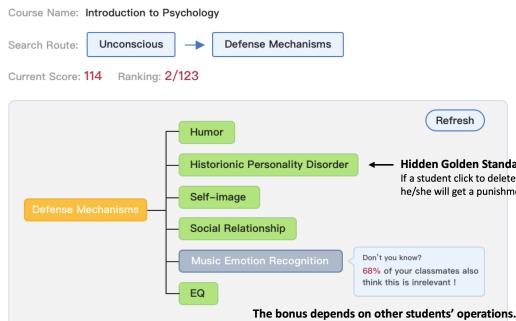


Figure 2: A demonstration of our interactive game in course *Introduction to psychology*. MOOC users can click irrelevant expansion candidates to get bonuses. The yellow concept on the left is from course, and the green concepts are expanded candidates.

Finally, we set up a game to present the expansion candidates. As shown in Figure 2, real MOOC users are drawn to pick out the *course-unrelated* ones to get bonuses. To ensure data quality, we set the game bonus depending on the group voting

<sup>3</sup>Anonymous for blind review.

result. We also avoid their irresponsible operations by mixing some extracted course concepts among candidates to detect the spoilers. The operation records are employed to train our reinforcement learning model proposed in the next section.

### 3 The Proposed Model

In this section, we first introduce our hierarchical reinforcement concept expansion framework, ExpanRL, then present our high-level seed selection model and low-level expansion model separately.

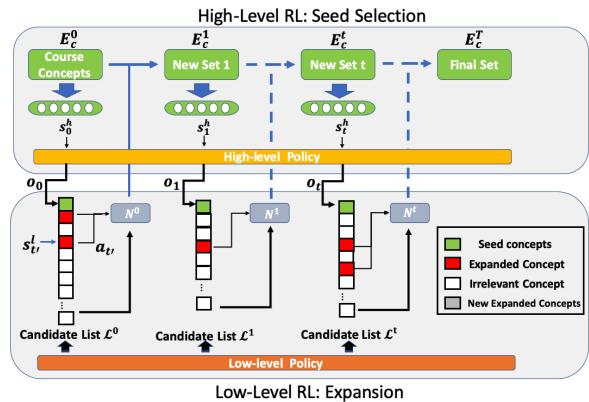


Figure 3: Framework of ExpanRL.

#### 3.1 Overview

To obtain high-quality expanded course concepts for serving students in MOOCs, ExpanRL still needs to address three crucial problems. 1. How to properly utilize user feedback? 2. How to keep accurate modeling of the course during iterations? 3. How to keep a good generalization ability of the model when expanding in new MOOC courses?

Thanks to the interactive MOOC environment, we can deal with these issues by decomposing the basic concept expansion workflow into a hierarchical reinforcement learning framework. Figure 3 shows that the model can learn the complex connection between concepts and courses from user feedback instead of simple contextual information. The main idea of ExpanRL is to upgrade expanding strategies via such an end-to-end model, whose entire expansion process works as the basic concept expansion methods in Section 2.2, which can be naturally formulated as a semi-Markov decision process (Sutton et al., 1999) like : 1) a high-level RL process that selects *seeds* from  $E_c^t$  to search for a list of candidates  $\mathcal{L}^t$ ; 2) a low-level RL process that detect the high-quality expansion results among candidates and obtain  $N^t$  to refresh the set

to  $E_c^{t+1}$ . This process iterate until the size of the expansion set reaches the preset limit,  $\tau$ .

Specially, before the whole process, we first utilize the method in (Pan et al., 2017) to extract course concepts  $\mathcal{M}$  from the given course corpus  $\mathcal{D}$  and initialize  $E_c^0 = \mathcal{M}$ .

### 3.2 Seed Selection with High-level RL

The high-level RL policy  $\mu$  aims to select  $k$  seeds from the existing set  $E_c$ , which can be regarded as a conventional RL over options. An option refers to a high-level action, and a low-level RL will be launched once the agent executes an option. The high-level time step  $t$  is the expansion round.

**Option:** The option  $o_t$  is a vector consisting of 0 and 1, which represents the  $i$ -th concepts from expansion set  $E_c^t$  is or is not a selected seed for the current expansion round. Thus the dimension of  $o_t$  is the same as the size of  $E_c^t$ . When a low-level RL process enters a final state, the agent’s control will be taken over to the high-level RL process to execute the next options.

**State:** The state  $s_t^h \in \mathcal{S}^h$  of the high level RL process at time step  $t$ , is represented by a  $k \times C$  matrix reshaped from the hidden state  $\mathbf{h}_t$ , where  $k$  is the size of seed set and  $C$  is the size of a compressed word embedding.

$$s_t^h = \text{reshape}(\mathbf{h}_t) \quad (1)$$

To obtain the hidden state  $\mathbf{h}_t$ , we introduce a set representation RepSet (Skianis et al., 2019) to encode the current expansion set  $E_c^t$ . RepSet is unsupervised, order independent and can encode an  $n \times V$  matrix to a  $V$  dimension vector. Note that  $E_c^{t-1} \subset E_c^t$ , so the current state is effected by the last state  $\mathbf{h}_{t-1}$ .

$$\mathbf{h}_t = \text{RepSet}(E_c^t). \quad (2)$$

**Policy:** The stochastic policy for seed selection  $\mu : \mathcal{S} \rightarrow \mathcal{O}$  which specifies a probability distribution over options:

$$o_t \sim \mu(o_t | s_t^h) = \mathbf{R}^t = \text{softmax}(\mathbf{s}_t^h \mathbf{W} (\mathbf{E}_c^t)^T). \quad (3)$$

where  $\mathbf{W}$  is a learnable parameter, which compresses a  $V$  length word embedding to a  $C$  length word embedding.  $\mathbf{E}_c^t$  is the matrix which consists of all course concepts’ word vector.  $\mathbf{R}^t$  is a matrix, while  $\mathbf{R}_{j,i}^t$  indicates the possibility of the  $i$ -th

concept in  $E_c^t$  to be the  $j$ -th seed:

$$p(K_j^t = c_i, c_i \in E_c^t | t) = \begin{cases} \mathbf{R}_{j,i}^t & \text{if } c_i \text{ is selected before.} \\ 0 & \text{otherwise.} \end{cases} \quad (4)$$

And the possibility of the high-level RL to select  $K^t$  is shown below. Note that this possibility  $p$  is independent of  $i$ .

$$p^h(K^t) = \prod_{j=1}^k p(K_j^t = c_i, c_i \in E_c^t | t) \quad (5)$$

**Reward:** Then, the environment provides intermediate reward  $r_t^h$  to estimate the future return when executing  $o_t$ . The reward is given by the total reward of the last round of concept expansion.

$$r_t^h = \sum r_{t'}^l(o_t), \quad (6)$$

where  $r_{t'}^l(o_t)$  is the low-level reward in time  $t'$  while the high-level option is  $o_t$ .

**Candidate generation after high-level options:** After the agent gives out an option  $o_t$ , we link the seed concepts from  $K^t$  into  $\mathcal{KB}$  and find their first-order neighbor concepts as the candidate list  $\mathcal{L}^t$ . Note that  $\mathcal{L}^t$  is sorted using the pairwise similarity between newly found candidates and seeds.

### 3.3 Concept Expansion with Low-level RL

Once the high-level policy has selected the seed set and generated a candidate list  $\mathcal{L}^t$ , the low-level policy  $\pi$  will scan the list and select high-quality expansion concepts from it to update  $E_c$ . The low-level policy over actions is formulated very similarly as the high-level policy over options. The option  $o_t$  and  $K^t$  from the high-level RL is taken as additional input throughout the low-level expansion process. The time step  $t'$  in low-level means the  $t'$ -th candidate in  $\mathcal{L}^t$  and the final expanded concepts in this round is  $N^t$ .

**Action:** The action at each time step is to assign a tag to the current candidate concept. The action space, i.e.,  $\mathcal{A} = \{1, 0\}$ , where 1 represents the present concept is an expansion result of this set, 0 represents that the concept is not an expansion result.

**State:** The low-level intra-option state  $s_t^l$  is represented by the word embedding of current expansion candidate  $c_{t'}$ .

$$s_t^l = \mathbf{c}_{t'} \quad (7)$$

Moreover, we use a Bi-LSTM (Huang et al., 2015) to provide a hidden state of current candidate list

$h_t^l$  by encoding: 1) the selected seeds  $K^t$ , 2) a zero vector as a segmentation, 3) the candidate list  $\mathcal{L}^t$ , thereby utilizing the information of high-level option  $o_t$  to help low-level decisions.

$$\mathbf{h}_t^l = BiLSTM([K^t; \mathbf{0}; \mathcal{L}^t]) \quad (8)$$

**Policy:** The stochastic policy for expansion  $\pi : \mathcal{S} \rightarrow \mathcal{A}$  outputs an action distribution given intra-option state  $\mathbf{s}_t^l$  and the high-level option  $o_{t'}$  that launches the current subtask. Here  $\odot$  is the vector dot product.

$$\begin{aligned} a_{t'} &\sim \pi(a_{t'} | \mathbf{s}_t^l; o_t) = p^l(c_{t'}) = p(c_{t'} \in N^t | t') \\ &= \text{sigmoid}(\mathbf{h}_t^l \odot \mathbf{s}_{t'}), \end{aligned} \quad (9)$$

**Reward:** As introduced in section of *Preliminaries*, we construct an interactive game on the MOOC website to collect feedback from users on the expanded concepts. Users can pick out the unrelated concepts of the course, and the picked times of each expansion result  $c_i$  is recorded as  $\varphi(c_i)$ . Since such operations indicate the users' disagreements of the result, the low-level reward is designed to be negatively correlated with  $\varphi(c_i)$  as follows:

$$r_{t'}^l = \begin{cases} -\varphi(c_i) / \max_{c_j \in \mathcal{L}^t} (\varphi(c_j)), a_{t'} = 1 \\ \varphi(c_i) / \max_{c_j \in \mathcal{L}^t} (\varphi(c_j)), a_{t'} = 0 \end{cases} \quad (10)$$

The count of user clicks determines the degree of relevance of each candidate to the course. It is worth noting that this degree is dynamic and depends on the concept that is mostly picked. This setting effectively controls the range of rewards.

**Set refreshment after low-level actions:** After the agent gives out an action  $a_{t'}$ , we can finally obtain the new expanded concepts  $N^t$ . The expansion set is updated as  $E_c^{t+1} = E_c^t \cup N^t$  and the process turn to another round.

### 3.4 Hierarchical Policy Learning

To optimize the high level policy, we aim to maximize the expected cumulative rewards from the main task at each step  $t$  as the agent samples trajectories following the high-level policy  $\mu$ , which can be computed as follows:

$$J(\theta_{\mu,t}) = \mathbb{E}_{\mathbf{s}^h, o, r^h \sim \mu(o|\mathbf{s}^h)} [\sum_{t=0}^T \log p^h(\mathbf{K}^t) \sum_{s=t}^T \gamma^{s-t} r_s^h], \quad (11)$$

where  $\mu$  is parameterized by  $\theta_\mu$ ,  $\gamma$  is a discount factor in RL, and the whole sampling process  $\mu$  takes  $T$  time steps before it terminates.

---

### Algorithm 1: Training Procedure of HRL

---

```

1 Extract course concepts from  $\mathcal{D}$  and initiate
    $E_c^0 = \mathcal{M}$ ;
2 Initiate state  $\mathbf{s}_0^h \leftarrow \mathbf{0}$  and time step  $t \leftarrow 0$ ;
3 while  $|E_c| < \sigma$  do
4   Calculate  $\mathbf{s}_t^h$  by Eq.(1);
5   Sample  $o_t$  from  $\mathbf{s}_t^h$  by Eq.(3);
6   Search for candidates from  $\mathcal{KB}$  and generate a
      ranked candidate list  $\mathcal{L}$ ;
7   for  $j \leftarrow 1$  to  $|\mathcal{L}|$  do
8      $t' \leftarrow t' + 1$ ;
9     Calculate  $\mathbf{s}_{t'}^l$  by Eq.(7);
10    Sample  $a_{t'}^l$  from  $\mathbf{s}_{t'}^l$  by Eq.(9);
11    Add the expansion result into game and get
        feedback;
12    Obtain low-level reward  $r_{t'}^l$  by Eq.(10);
13  end
14   $t \leftarrow t + 1$ , refresh  $E_c$ ;
15  Obtain low-level final reward  $r_{fin}^l$ , high-level
     reward  $r_t^h$ ;
16 end
17 Obtain high-level final reward  $r_{fin}^h$  by Eq.(6);
18 Optimize the model with Eq.(1) and Eq.(12);

```

---

Similarly, we learn the low-level policy by maximizing the expected cumulative intra-option rewards from the sub task over option  $o_t$  when the agent samples along low-level policy  $\pi(\cdot | o_t)$  at time step  $t$ :

$$J(\theta_{\pi,t}; o_{t'}) = \mathbb{E}_{\mathbf{s}^l, a, r^l \sim \pi(a|\mathbf{s}^l; o_{t'})} [\sum_{t'=0}^{T'} \log p^l(\mathbf{c}^{t'}) \sum_{s=t'}^{T'} \gamma^{s-t} r_s^l], \quad (12)$$

if the subtask ends at time step  $T'$ .

Then we use policy gradient methods (Sutton et al., 2000) with the REINFORCE (Williams, 1992) algorithm to optimize both high-level and low-level policies. The entire training process is described at Algorithm 1.

## 4 Experiments

### 4.1 Experiment Setting

#### 4.1.1 Datasets

We construct an interactive MOOC environment as Section 2.3 to collect user feedback on expansion results. To build a solid evaluation, we randomly selected 5% expanded concepts to be manually labeled benchmarks. For each concept, three annotators majoring in the corresponding domain are asked to label them as “0: Not helpful” or “1: Helpful” based on their knowledge. Thus, each dataset is triply annotated, and Pearson correlation coefficient is computed to assess the inter-annotator agreement. A candidate is labeled as a related concept when more than two annotators give positive

	<b>MAT</b>	<b>CHEM</b>	<b>PSY</b>	<b>MS</b>	<b>ARC</b>	<b>CS</b>	<b>MAT+CS</b>	<b>CHEM+MS</b>	<b>MS+ARC</b>
<b>#courses</b>	12	6	16	8	14	12	4	4	5
$ \mathcal{M} $	1,688	1,404	568	842	1,036	1,015	230	417	382
<b>#operations</b>	93,762	103,652	48,492	40,254	120,384	88,779	33,521	52,467	56,787
<i>0-Label</i>	24,278	15,796	13,245	11,876	33,127	17,775	7,092	9,367	7,898
<i>1-Label</i>	6,976	18,755	2,919	1,542	7,001	11,818	3,533	4,790	1,229
<i>correlation</i>	0.712	0.694	0.705	0.732	0.678	0.689	0.655	0.688	0.701

Table 1: Statistics of datasets

tags. Table 1 presents the detailed statistics, where *#courses*,  $|\mathcal{M}|$ , *1-Label* and *0-Label* are the number of courses, course concepts, positive and negative labels. *#operations* are user click times which is obtained from the game. *MAT*, *CHEM*, *PSY*, *MS*, *ARC* and *CS* correspond to Mathematics, Chemistry, Psychology, Material Science, Architecture and Computer Science.

In particular, we select 13 interdisciplinary courses<sup>4</sup> and build three multi-category course datasets as *MAT+CS*, *CHEM+MS* and *MS+ARC* to further estimate the performance of ExpanRL on interdisciplinary courses. Note that these three datasets are subsets of the above six’s.

**Dataset Usage.** All the models are trained on the user operation data and evaluated on the expert annotated data. For the supervised learning baselines, we set the concepts with top 70% click records as negative, and the rest as positive samples.

#### 4.1.2 Basic Settings

All hyper-parameters are tuned on the validation set. The dimension of word vectors in Eq. (2) is 768. The dimension of the compressed word vector  $C$  in Eq. (1) is 128. The word vectors of all baseline methods are initialized using BERT (Devlin et al., 2019). The learning rate is  $1.0 \times 10^{-4}$  for low-level RL, and  $1.0 \times 10^{-5}$  for high-level. The discount factor  $\gamma$  is 0.99. The seed size  $k$  is set to 10 and the upper limit  $\tau$  of  $E_c$  is 20,000.

#### 4.1.3 Baselines

We compare our hierarchical RL model (denoted as HRL) with five typical methods of set expansion. As these methods obtain expansion candidates from diverse resources, we mainly employ the different similarity metrics to rank the same expansion candidate list for evaluation. Especially to investigate the impact of seed selection strategies, we use a K-means clustering-based method and a pairwise similarity-based method to replace the high-level RL network, which are denoted as C-RL and P-RL.

<sup>4</sup>Course list is shown in Appendix.

- **PR.** Graph based method: We build the candidates and course concepts into a graph. When the similarity between two concepts exceeds a threshold<sup>5</sup>  $\sigma_{PR}$ , there is a link between them. The PageRank score of each candidate is finally used for sorting. A most famous method employing graph based ranking is SEAL (Wang and Cohen, 2007)

- **SEISA.** SEISA (He and Xin, 2011) is an entity set expansion system developed by Microsoft after SEAL and outperforms traditional graph-based methods by an original unsupervised similarity metric. We implement its Dynamic Thresholding algorithm to sort expanded concepts.

- **EMB.** Embedding based method mainly utilizes context information to examine the similarity between expanded concepts and seeds according to (Mamou et al., 2018). For each expanded concept  $e$ , we calculate the sum of its cosine similarities with course concepts  $\mathcal{M}$  in BERT (Devlin et al., 2019) and use the average as golden standard to rank the expanded concept list.

- **PUL.** PU learning is a semi-supervised learning model regarding set expansion as a binary classification task. We employ the same setting as (Wang et al., 2017) to classify and sort concepts.

- **PIP.** It is a pipeline method for course concept expansion (Yu et al., 2019a), which first uses an online clustering method during candidate generation and then classify them to obtain final expansion results. We follow the workflow of this work to sort expanded concepts.

#### 4.1.4 Evaluation Metrics

Our objective is to generate a ranked list of expanded concepts. Thus, we use the **Mean Average Precision**(MAP) as our evaluation metric, which is the preferred metric in information retrieval for evaluating ranked lists.

#### 4.2 Overall Evaluation

Table 2 summarizes the comparing results of different methods on all datasets. The evaluation is

<sup>5</sup> $\sigma_{PR}$  is experimentally set to 0.5.

	<b>MAT</b>	<b>CHEM</b>	<b>PSY</b>	<b>MS</b>	<b>ARC</b>	<b>CS</b>	<b>Avg</b>	<b>MAT+CS</b>	<b>CHEM+MS</b>	<b>MS+ARC</b>	<b>I-Avg</b>
PR	0.763	0.705	0.482	0.470	0.300	0.690	0.568	0.659	0.664	0.401	0.575
SEISA	0.805	0.711	0.473	0.524	0.570	0.713	0.632	0.797	0.691	0.377	0.622
EMB	0.747	0.687	0.474	0.533	0.442	0.812	0.616	0.710	0.655	0.377	0.581
PUL	0.878	0.811	0.845	0.745	0.757	0.850	0.822	0.880	0.782	0.646	0.769
PIP	0.848	0.782	0.803	0.772	0.775	0.821	0.800	0.893	0.835	0.851	0.865
C-RL	0.902	0.795	0.818	0.753	0.716	0.800	0.797	0.851	0.849	0.758	0.820
P-RL	0.892	0.768	0.606	0.749	0.821	0.767	0.835	0.871	0.852	0.662	0.795
HRL	<b>0.903</b>	<b>0.857</b>	<b>0.901</b>	<b>0.806</b>	<b>0.828</b>	<b>0.878</b>	<b>0.862</b>	<b>0.909</b>	<b>0.903</b>	<b>0.886</b>	<b>0.898</b>

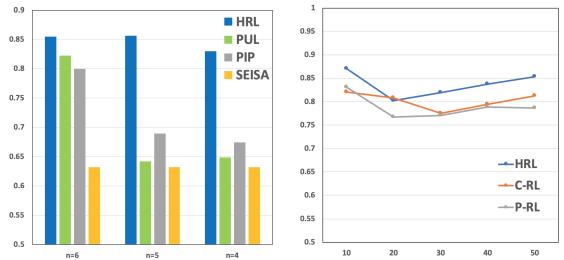
Table 2: MAP of different methods on datasets. (Seed set size = 10)

divided into two parts. The six datasets on the left are the performance of the model on various subjects, and **Avg** represents the average of their MAPs. The three datasets on the right are from the selected interdisciplinary courses, and **I-Avg** is the average of the model performance on them. We also divide the methods into unsupervised, supervised, and reinforcement learning models for further analysis. Overall, our approach HRL maintains an impressive performance (at 0.862 of Avg and 0.898 of I-Avg) over the existing methods, and unsupervised methods (such as SEISA, PR) are not so competitive when compared with methods with supervised information. We lead a detailed investigation to detect the performance among different datasets and the impact of seed selection in the following aspects:

**For different datasets**, our methods achieve robust results. It is worth noting that the range of the MAP of our method on these datasets does not exceed 0.097, while other baselines suffering from severe oscillations (SEISA of 0.428, EMB of 0.435, and PUL of 0.234). And these supervised methods (PUL, PIP) that perform well on a certain dataset are further analyzed in subsequent experiments.

**For the performance on interdisciplinary courses.** Most of the baselines meet a decline when turned to interdisciplinary courses. From this angle, PUL can not face this challenge. But PIP, C-RL, and HRL perform even better (with a lift of 0.04 on average), most likely because they all have a clustering-like seed selection process.

**For different seed selection strategies.** We also detect the impact of seed selection by replacing high-level RL. The comparison among three RL methods shows that: 1) P-RL performs better in one-category expansion tasks (beat C-RL at 0.038); 2) C-RL deal with interdisciplinary courses better than P-RL (as discussed above); 3) HRL is stronger than these two methods in all datasets. The results



(a) The MAP of different number of training sets.

Figure 4: Performance of different settings. (a) shows the average MAP when mask some of the datasets in training. (b) shows the MAP of different seed size.

exactly prove the superiority of HRL’s seed selection over rule-based strategies.

### 4.3 Result Analysis

**Generalization Ability.** Expansion models in MOOCs need to face with plenty of new courses every day. Thus we lead strict experiments to estimate the generalization ability of the model by masking training datasets. For example, the bar of  $n = 5$  in Figure 4(a) indicates the average MAP when the models are trained on five subject datasets and tested on the other one. Thus  $n = 6$  is the average MAP in Table 2 while  $n = 5$  and  $n = 4$  present the results of facing one or two kinds of new courses. Here we select HRL, PUL, and PIP for observation. Such an experiment shows that HRL still maintains an outstanding performance in new courses. Still, PIP and PUL suffer from a sharp decline in untrained new datasets (even at the same level as unsupervised methods).

**The size of seed set  $k$ .** For different settings of seed sizes, we compare the performance of ExpanRL with other RL based baselines. As shown in Figure 4(b), HRL keeps a high level of MAP among these settings (all over 0.8 on average). Meanwhile, we find that all these RL-based methods perform

	<b>Cr@10</b>	<b>Cr@20</b>	<b>Cr@50</b>
PR	0.097	0.182	0.425
SEISA	0.097	0.204	0.459
EMB	0.071	0.150	0.359
PUL	0.041	0.091	0.349
PIP	0.069	0.126	0.342
HRL	<b>0.036</b>	<b>0.082</b>	<b>0.258</b>

Table 3: Online Evaluation results.

better in small or large seed size (less than 10 or larger than 40), which requires future detection on this phenomenon.

**Discussion.** Based on the above experimental results, we summarize the analysis as follows: 1) the performance of unsupervised methods on different datasets is not as stable as the supervised or RL methods; 2) except for models that have a clustering-like seed selection process (PIP, CRL, HRL), most models suffer from declines on interdisciplinary datasets; 3) although supervised models (PIP, PUL) perform well in some cases, they drastically decline in untrained new courses; 4) HRL, consisting of a feasible seed selection RL and expansion strategies from human efforts, keep a high performance under different settings. HRL deal with the challenges in MOOC expansion tasks, as claimed in the introduction.

#### 4.4 MOOC Online Evaluation

Utilizing user feedback on the expansion results from our interactive MOOC environment, we also set up an online evaluation to detect whether users agree on the expansion results. Following the same evaluation metric in (Yu et al., 2019a), we denote **Click Rate** as  $C_r@q$ , which means the click rate of top  $q$  expanded concepts, i.e.,

$$C_r@q = \sum_{i=1}^q \varphi(c_i) / \sum_{j=1}^{|E_c|} \varphi(c_j) \quad (13)$$

A smaller  $C_r@q$  indicates more users think the results are relevant to the course. We record the performance of each method in Table 3. Results show that ExpanRL obtains the best feedback from MOOC users under all three settings. It's worth noting that the advantage of ExpanRL is evident while selecting larger-scale samples (The overlap rises from 0.005 to 0.091), which indicates that our model can provide more high-quality concepts.

#### 5 Related Work

Our work follows the task of concept expansion in MOOCs (Yu et al., 2019a), a particular type of set

expansion problem, which takes several seeds as input and expands the entity set.

Set expansion was born to serve knowledge acquisition applications on the Internet. Google Sets was a pioneer which leaded a series of early research, e.g. Bayesian Sets (Ghahramani and Heller, 2006), SEAL (Wang and Cohen, 2007), SEISA (He and Xin, 2011) and others (Sarmento et al., 2007; Shi et al., 2010; Wang et al., 2015). These efforts utilize web tables as a resource and mainly serves for search engines. Recently, more related research has turned its attention to other application fields, such as news mining (Redondo-García et al., 2014), knowledge graphs (Zhang et al., 2017), education assistance (Yu et al., 2019a), etc. Meanwhile, corpus-based expansion methods snowball, and iterative bootstrapping became a common solution (Shen et al., 2017; Yu et al., 2019b; Yan et al., 2019), which expands the set in round and select high-quality results to extract feature iteratively. ExpanRL is inspired by this type of method and is designed to optimize the existing iterative process.

ExpanRL also benefits from hierarchical reinforcement learning (HRL), which has been employed in many NLP tasks (Zhang et al., 2019; Takanobu et al., 2019) and achieved impressive results. By decomposing complex tasks into multiple small tasks to reduce the complexity of decision making (Barto and Mahadevan, 2003), HRL naturally matches the iterative set expansion tasks.

#### 6 Conclusion and Future Work

We investigate the task of course concept expansion, which utilizes the NLP approaches in improving MOOC education. After constructing a novel interactive MOOC environment to collect user feedback on expansion results, we design a paradigm, ExpanRL, which decomposes the concept expansion task into a hierarchy of two subtasks: high-level seed selection and low-level concept expansion. Experiment results on nine datasets from real MOOCs prove that ExpanRL can better serve students by recognizing the helpful expanded results and maintaining good performance in interdisciplinary courses and even new courses.

Promising future directions include detecting how to ensemble supervised learning and RL expansion models and applying the proposed model in related tasks. We also hope our design of interactive games can call for more fancy methods that utilize student feedback in NLP applications

in Education.

## Acknowledgement

This work is supported by the National Key Research and Development Program of China (2018YFB1004503), NSFC Key Projects (U1736204, 61533018), grants from Beijing Academy of Artificial Intelligence (BAAI2019ZD0502), Institute for Guo Qiang, Tsinghua University (2019GQB0003), and XuetangX.

## References

- Ausubel. 1968. Educational psychology: A cognitive view.
- Andrew G Barto and Sridhar Mahadevan. 2003. Recent advances in hierarchical reinforcement learning. *Discrete event dynamic systems*, 13(1-2):41–77.
- James R Curran, Tara Murphy, and Bernhard Scholz. 2007. Minimising semantic drift with mutual exclusion bootstrapping. In *Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics*, volume 6, pages 172–180.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- Zoubin Ghahramani and Katherine A Heller. 2006. Bayesian sets. In *Advances in neural information processing systems*, pages 435–442.
- Yeye He and Dong Xin. 2011. Seisa: set expansion by iterative similarity aggregation. In *Proceedings of the 20th international conference on World wide web*, pages 427–436.
- Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.
- Hailong Jin, Chengjiang Li, Jing Zhang, Lei Hou, Juanzi Li, and Peng Zhang. 2019. Xlore2: Large-scale cross-lingual knowledge graph construction and application. *Data Intelligence*, 1(1):77–98.
- Jonathan Mamou, Oren Pereg, Moshe Wasserblat, Alon Eirew, Yael Green, Shira Guskin, Peter Izsak, and Daniel Korat. 2018. Term set expansion based nlp architect by intel ai lab. *EMNLP 2018*, page 19.
- Liangming Pan, Xiaochen Wang, Chengjiang Li, Juanzi Li, and Jie Tang. 2017. Course concept extraction in moocs via embedding-based graph propagation. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 875–884.
- Pushpendre Rastogi, Adam Poliak, Vince Lyzinski, and Benjamin Van Durme. 2019. Neural variational entity set expansion for automatically populated knowledge graphs. *Information Retrieval Journal*, 22(3-4):232–255.
- José Luis Redondo-García, Michiel Hildebrand, Lilia Perez Romero, and Raphaël Troncy. 2014. Augmenting tv newscasts via entity expansion. In *European Semantic Web Conference*, pages 472–476. Springer.
- Luis Sarmento, Valentin Jijkuon, Maarten De Rijke, and Eugenio Oliveira. 2007. More like these: growing entity classes from seeds. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 959–962.
- D Shah. 2019. Year of mooc-based degrees: A review of mooc stats and trends in 2018. class central. *Class Central’s MOOC Report*.
- Jiaming Shen, Zeqiu Wu, Dongming Lei, Jingbo Shang, Xiang Ren, and Jiawei Han. 2017. Setexpan: Corpus-based set expansion via context feature selection and rank ensemble. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 288–304.
- Jiaming Shen, Zeqiu Wu, Dongming Lei, Chao Zhang, Xiang Ren, Michelle T Vanni, Brian M Sadler, and Jiawei Han. 2018. Hiexpan: Task-guided taxonomy construction by hierarchical tree expansion. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2180–2189.
- Shuming Shi, Huibin Zhang, Xiaojie Yuan, and Ji-Rong Wen. 2010. Corpus-based semantic class mining: distributional vs. pattern-based approaches. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 993–1001.
- Konstantinos Skianis, Giannis Nikolentzos, Stratis Limnios, and Michalis Vazirgiannis. 2019. Rep the set: Neural networks for learning set representations. *arXiv preprint arXiv:1904.01962*.
- Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063.
- Richard S Sutton, Doina Precup, and Satinder Singh. 1999. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211.

Ryuichi Takanobu, Tianyang Zhang, Jiexi Liu, and Minlie Huang. 2019. A hierarchical framework for relation extraction with reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7072–7079.

Chi Wang, Kaushik Chakrabarti, Yeye He, Kris Ganjam, Zhimin Chen, and Philip A Bernstein. 2015. Concept expansion using web tables. In *Proceedings of the 24th International Conference on World Wide Web*, pages 1198–1208.

Richard C Wang and William W Cohen. 2007. Language-independent set expansion of named entities using the web. In *Seventh IEEE international conference on data mining (ICDM 2007)*, pages 342–350. IEEE.

Yasheng Wang, Yang Zhang, and Bing Liu. 2017. Sentiment lexicon expansion based on neural pu learning, double dictionary lookup, and polarity association. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 553–563.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.

Lingyong Yan, Xianpei Han, Le Sun, and Ben He. 2019. Learning to bootstrap for entity set expansion. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 292–301, Hong Kong, China. Association for Computational Linguistics.

Jifan Yu, Chenyu Wang, Gan Luo, Lei Hou, Juanzi Li, Zhiyuan Liu, and Jie Tang. 2019a. Course concept expansion in moocs with external knowledge and interactive game. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 4292–4302.

Puxuan Yu, Zhiqi Huang, Razieh Rahimi, and James Allan. 2019b. Corpus-based set expansion with lexical features and distributed representations. In *Proceedings of the 42Nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR’19*, pages 1153–1156.

Jing Zhang, Bowen Hao, Bo Chen, Cuiping Li, Hong Chen, and Jimeng Sund. 2019. Hierarchical reinforcement learning for course recommendation in moocs. *Psychology*, 5(4.64):5–65.

Xiangling Zhang, Yueguo Chen, Jun Chen, Xiaoyong Du, Ke Wang, and Ji-Rong Wen. 2017. Entity set expansion via knowledge graphs. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR’17*, pages 1101–1104. ACM.

## A Dataset Analysis & Case Study

We also analyze the characteristics of the datasets and do a case study to explore further the impact of different expansion tasks on the model, which will help choose the appropriate expansion model for various tasks.

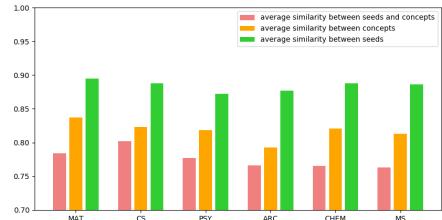


Figure 5: The average pairwise similarity of seeds, expanded concepts and seed-expand concept pairs.

We assess the degree of dispersion of the concepts from different subjects by calculating the pairwise average similarities. Combining the results in Overall Evaluation and Figure 5, we find the science subjects, MAT and CHEM, obtain the most aggregated concepts (Green bars in Figure), which also leads to a booming of all methods in this two datasets. Simultaneously, unsupervised models (SEISA, EMB) show significant performance degradation on PSY and ARC datasets, with the lowest average similarity of expansion results (Red and orange bars). This demonstrates the critical role of supervisory information in complex set expansion.

The contest between the supervised learning methods (PUL, PIP) and the RL methods can be observed more intuitively through the case study. We sample some errors from ARC and CHEM datasets in Figure 6. It is easy to find that the errors of supervised learning methods mainly come from some noise words, e.g., the word “architecture” in *computer architecture*. However, the errors of RL methods are mainly caused by classification, e.g., *electric potential energy* is highly relevant to chemistry, but it is a physics concept.

From this phenomenon, we speculate that SL knows more about the context of the concept, and RL understands the meaning of the concept better. Therefore, the joint method of combining supervised learning and RL is likely to be a promising research direction in expansion tasks.

	False Positive	False Negative
Chemistry	<i>Computer architecture</i> <i>Astronomical</i>	<i>Glass</i> <i>Richard B. Fuller</i> <i>Rubber bearings</i>
Architecture	<i>Matrix factorization</i> <i>Electric potential energy</i>	<i>Claude Ruggieri</i> <i>Resonance enhancement</i>

Figure 6: Some error cases in ARC and CHEM datasets. Blue concepts are errors from supervised methods, orange ones are from RL methods and black is the shared errors.

## B List of interdisciplinary courses

In this section, we list the selected interdisciplinary courses to present this situation in real MOOCs. As shown in Table 4, many courses from MOOCs are related to more than one subject; this is a common phenomenon in practical teaching. The URLs of these courses are hidden for blind review.

Domain	CourseName
MAT+CS	Introduction to Data Science
	Computational Geometry
	Algorithm of Big Data
	Multivariate statistical analysis and R language modeling
CHEM+MS	Plant Fiber Chemistry
	Chemical Reaction Engineering
	Magical Material World
	Catalyst Design and Preparation
MS+ARC	Construction Materials
	Architecture Materials
	Explore the Materials Around You
	Road Engineering Materials
	Reinforced Concrete and Masonry Structures

Table 4: The list of selected interdisciplinary courses.

# Vocabulary Matters: A Simple yet Effective Approach to Paragraph-level Question Generation

Vishwajeet Kumar

IITB-Monash Research Academy

Mumbai, India

vishwajeet@cse.iitb.ac.in

Manish Joshi

IIT Bombay

Mumbai, India

joshimanish0511@gmail.com

Ganesh Ramakrishnan

IIT Bombay

Mumbai, India

ganesh@cse.iitb.ac.in

## Abstract

Question generation (QG) has recently attracted considerable attention. Most of the current neural models take as input only one or two sentences and perform poorly when multiple sentences or complete paragraphs are given as input. However, in real-world scenarios, it is very important to be able to generate high-quality questions from complete paragraphs. In this paper, we present a simple yet effective technique for answer-aware question generation from paragraphs. We augment a basic sequence-to-sequence QG model with dynamic, paragraph-specific dictionary and copy attention that is persistent across the corpus, without requiring features generated by sophisticated NLP pipelines or handcrafted rules. Our evaluation on SQuAD shows that our model significantly outperforms current state-of-the-art systems in question generation from paragraphs in both automatic and human evaluation. We achieve a 6-point improvement over the best system on BLEU-4, from 16.38 to 22.62.

## 1 Introduction and Related work

Automatic question generation (QG) from text aims to generate meaningful, relevant, and answerable questions from a given textual input. Owing to its applicability in conversational systems such as Cortana, Siri, chatbots, and automated tutoring systems, QG has attracted considerable interest in both academia and industry. Recent neural network-based approaches (Du et al., 2017; Kumar et al., 2018a,b; Du and Cardie, 2018; Zhao et al., 2018; Song et al., 2018; Subramanian et al., 2018; Tang et al., 2017; Wang et al., 2017) represent the state-of-the-art in question generation. Most of these techniques learn to generate questions from short text, *i.e.*, one or two sentences (Du et al., 2017; Kumar et al., 2018a,b; Du and Cardie, 2018). On the other hand, the ability to generate high-quality questions from longer text such as from multiple

Yuan-Fang Li

Monash University

Melbourne, Australia

yuanfang.li@monash.edu

sentences or from a paragraph in its entirety, is more useful in real-world settings. However, given that a paragraph contains a longer context and more information than a sentence, it is a significantly more challenging problem to generate questions around a longer context. In figure 1 we present one motivating example demonstrating why the model needs information more than just a single sentence for generating question a meaningful and relevant question. As we can see in figure 1, question 2, question generated by our model use multiple sentences as context. Du et al. (2017) recently observed that 20% of the questions in the SQuAD dataset (Rajpurkar et al., 2016) require paragraph-level information to answer them. For the same reason, it is intuitive to conclude that the ability to consider the complete context; however long it may be, is critical for generating high-quality questions.

Legislative power in Warsaw is vested in a unicameral Warsaw City Council ( Rada Miasta ), which comprises 96 members . Council members are elected directly every four years . Like most legislative bodies , the City Council divides itself into committees which have the oversight of various functions of the city government . Bills passed by a simple majority are sent to the mayor ( the President of Warsaw ), who may sign them into law . If the mayor vetoes a bill , the Council has 30 days to override the veto by a two-thirds majority vote .

Human Generated:	How many members are on the Warsaw City Council ?
Our Model:	How many members are in the Warsaw City Council ?
Human Generated:	How often are elections for the counsel held ?
Our Model:	How often are the Rada Miasta elected ?
Human Generated:	What does the City Council divide itself into ?
Our Model:	The City Council divides itself into what ?
Human Generated:	How many days does the Council have to override the mayor 's veto ?
Our Model:	How long does it take to override the veto ?

Figure 1: Examples of ground-truth questions and questions generated by our model from the same paragraph. Each question and its corresponding answer are highlighted using the same color.

Zhao et al. (2018) very recently proposed a technique (referred to MPGNN here) for paragraph-level question generation using a max out pointer mechanism and a gated self-attention encoder. Their best model achieves BLEU-4 of 16.38 on SQuAD with paragraphs as input. Compared to (Zhao et al.,

2018), our model has less number of parameters (making it more computationally efficient), is relatively easy to train and is somewhat deterministically biased toward the generation of important words in the input paragraph.

In this paper, we propose a simple yet effective paragraph-level question generation technique. We augment the standard sequence-to-sequence model based on bidirectional LSTM with two components: (1) a dynamic, paragraph-specific dictionary and (2) a copy attention mechanism that is persistent across paragraphs. Our evaluation on SQuAD shows significant improvement over MPGNS in automatic evaluation. We achieve a 6-point increase with respect to BLEU-4 (from 16.38 to 22.62) over MPGNS’s best system. We perform the human evaluation of our model with and without copy attention, and we observe that we obtain 27% more relevant questions when the copy attention is incorporated.

For a given paragraph as input, we depict in Figure 1, the ground-truth questions as well as the questions generated along with the answers highlighted in the paragraph. As can be seen from the example, while generating the second question(highlighted in green color), our model uses information not only from the sentence containing the answer, but also relevant context from the complete paragraph.

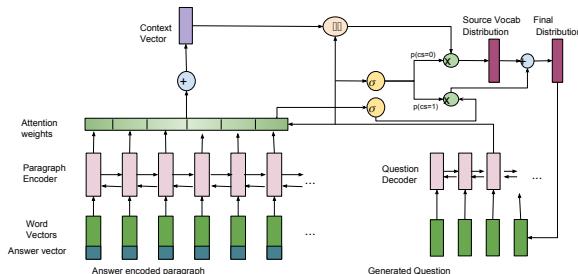


Figure 2: Overall architecture of our paragraph-level question generation model.

## 2 Problem Formulation & Approach

Given a paragraph ‘ $P$ ’ and answer ‘ $A$ ’, a question generation model iteratively samples question word  $q_t \in V^Q$  at every time step ‘ $t$ ’ from the probability distribution given by:

$$\Pr(Q|P,A;\theta) = \prod_{t=1}^{|Q|} \Pr(q_t|P,A;\theta) \quad (1)$$

Where  $V^Q$  is the question vocabulary,  $\theta$  is the set of parameters, and  $A$  is the answer.

Our question generation model consists of a two-layer paragraph encoder and a one-layer question decoder, equipped with a dynamic dictionary and copy attention. In Figure 2, we illustrate the overall architecture of our paragraph level question generation model. The dynamic dictionary allows every training instance (paragraph) to have its own vocabulary instead of relying on the preprocessed global vocabulary. Copy attention enables the model to predict question words from the extended vocabulary (complete vocabulary + paragraph vocabulary). Copy attention operates over the union of words in vocabulary and paragraph words.

### 2.1 Paragraph encoder

We use a two-layer bidirectional long short-term memory (Bi-LSTM) network stack as the paragraph encoder. The paragraph encoder takes an answer-tagged paragraph as input and outputs a representation of the paragraph. Note that the Bi-LSTM network processes the input paragraph in both the forward and backward directions:  $\vec{h}_t = \text{LSTM}(e_t, \vec{h}_{t-1})$  and  $\overleftarrow{h}_t = \text{LSTM}(e_t, \overleftarrow{h}_{t+1})$ , where  $\vec{h}_t$  (resp.  $\overleftarrow{h}_t$ ) is the forward (resp. backward) hidden state at time step  $t$  and  $e_t$  is the vector representation of current input  $x_t$  at time step  $t$ . The final hidden state for the current word input is the concatenation of the forward and backward hidden state vectors:  $h_t = [\vec{h}_t, \overleftarrow{h}_t]$ .

### 2.2 Dynamic, shared dictionary

In the traditional approach, a new/unknown word is typically replaced with the “`<unk>`” token. The copy mechanism (Gu et al., 2016) then unfortunately learns to copy this “`<unk>`” token instead of the actual (unknown) word from the source paragraph. Instead, we use a separate dynamic dictionary unique to each source paragraph, which includes all and only words that occur in the paragraph. This allows our model to copy source words that may not be in the target dictionary into the target (question). Using a dynamic dictionary consisting of the preprocessed vocabulary instead of a static one enables the copy mechanism to copy the exact words directly into the question, even if they are rare and unknown.

Given a source paragraph  $p$ , we denote its dynamic vocabulary by  $V^p$ . Our copy attention mechanism takes into account  $V^p$  and the global vocabulary  $V$  to determine whether to copy a word from  $V^p$  or to predict a word from question vocabulary  $V^Q$ .

As our model’s source as well as target are in

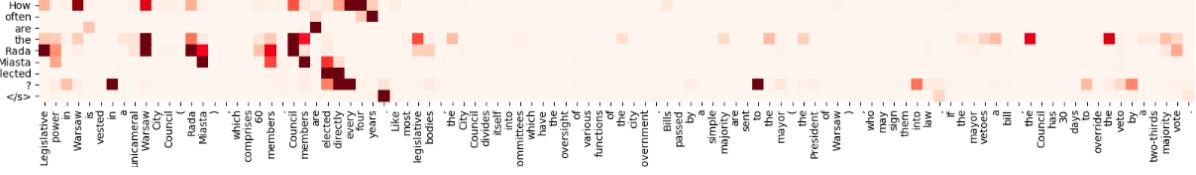


Figure 3: Visualizing attention weights for the second generated question in Fig. 1.

the same language, we work with a shared source and target vocabulary, though we learn different language models for the paragraph and the question. Sharing source and target vocabulary also decreases the memory requirement resulting from matrix multiplication (thus making faster training through larger batch size) possible. It also enables efficient question decoding, thus reducing the time for inference on the test data.

### 2.3 Question decoder

Our question decoder is another Bi-LSTM that takes as input the last hidden state and context representation from the encoder and generates question words sequentially based on the previously generated words. The decoder hidden state ( $s_t = [\vec{s}_t, \overleftarrow{s}_t]$ ) at time step  $t$  is the concatenation of the forward and backward hidden state representations:  $\vec{s}_t = LSTM(o_t, \vec{s}_{t-1})$  and  $\overleftarrow{s}_t = LSTM(o_t, \overleftarrow{s}_{t+1})$ , where  $o_t$  is the vector representation of decoder input ( $y_t$ ) at time step  $t$ . During training time the vector representation of words from the ground-truth question is fed as decoder input, and during test time the vector representation of the vocabulary word with maximum probability is fed as input. We feed **EOS** symbol as input to decoder from both forward and backward direction at time  $t_0$ . Bidirectional decoder factorizes the conditional decoding probabilities in both directions (left-to-right and right-to-left) into summation as:

$$P(y_t | [y_m]_{m \neq t}) = \frac{\log p(y_t | Y_{[1:t-1]})}{\log P(y_t | Y_{[t+1:T_y]})} \quad (2)$$

The probability distribution over words in the vocabulary is calculated as:

$$\Pr(q_t) = softmax(\mathbf{W}_g \sigma(\mathbf{W}_s [s_t, h_t] + \mathbf{b}_s) + \mathbf{b}_g) \quad (3)$$

where  $\mathbf{W}_g$ ,  $\mathbf{W}_s$ ,  $\mathbf{b}_s$  and  $\mathbf{b}_g$  are trainable model parameters. Probability distribution  $P(q_t)$  uses the standard softmax over the question vocabulary  $V^Q$ . This is used to sample word with maximum probability while decoding a question.

### 2.4 Copy attention

We know that a good question should be relevant to (answerable from) the paragraph. So we learn a probabilistic mixture model over the question vocabulary  $V^Q$  and the current paragraph vocabulary  $V^P$ . The current paragraph vocabulary is generated by a dynamic dictionary module.

Our copy attention calculates two values:

$cs$ : a binary-valued variable which acts a switch between copying a word from the paragraph's dynamic vocabulary  $V^P$  or generating from the question vocabulary  $V^Q$

$\Pr(.|V^P)$ : probability of copying a particular word from paragraph vocabulary  $V^P$ .

Therefore, the final probability distribution from which a word will be sampled while generating a question is calculated over the extended vocabulary  $V^Q \cup V^P$ . Given a word from the extended vocabulary  $w \in V^Q \cup V^P$ , its probability  $\Pr(w)$  is computed as:

$$\Pr(w) = \Pr(cs=1)\Pr(w|V^P) + \Pr(cs=0)\Pr(w|V^Q) \quad (4)$$

The switch probability  $\Pr(cs)$  is determined using the decoder hidden states as:

$$\Pr(cs=1) = \sigma(\mathbf{W}_{cs} s_t + \mathbf{b}_{cs}) \quad (5)$$

where  $\mathbf{W}_{cs}$  and  $\mathbf{b}_{cs}$  are trainable model parameters.  $\Pr(w|V^Q)$  is the probability of predicting a word from complete vocabulary  $V^Q$ . The copy attention weight  $a^t$  is computed as:

$$e_i^t = v^T \tanh(\mathbf{W}_h h_i + \mathbf{W}_s s_t + b_{attn}) \quad (6)$$

$$a^t = sparsemax(e^t) \quad (7)$$

Where  $v$ ,  $\mathbf{W}_h$ ,  $\mathbf{W}_s$  and  $b_{attn}$  are trainable model parameters. The probability of copying a word from the paragraph vocabulary  $V^P$  is estimated as:

$$\Pr(w|V^P) = \sigma(\mathbf{W}_a a^t + \mathbf{b}_a) \quad (8)$$

where  $\mathbf{W}_a$  and  $\mathbf{b}_a$  are trainable model parameters.

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE-L
MPGSN (Zhao et al., 2018)	45.07	29.58	21.60	16.38	20.25	44.48
L2A (Du et al., 2017)	42.54	25.33	16.98	11.86	16.28	39.37
NQG <sub>dd</sub> [w/o copy attention]	55.32	32.39	20.12	12.86	17.00	42.77
NQG <sub>dd</sub> [with copy attention]	<b>61.84</b>	<b>41.73</b>	<b>30.19</b>	<b>22.62</b>	<b>21.93</b>	<b>48.60</b>

Table 1: Results on the test set on automatic evaluation metrics. Best results for each metric (column) are **bolded**.

### 3 Experimental Setup

We report the experimental result of our model (referred to as NQG<sub>dd</sub>) and compare it with the current state of the art MPGSN (Zhao et al., 2018). We employ the widely-used metrics BLEU (Papineni et al., 2002), ROUGE-L and METEOR for automatic evaluation. We use evaluation script provided by (Chen et al., 2015). Similar to (Kumar et al., 2018a) we also report qualitative assessment on the syntax, semantics and relevance of the questions generated by our model.

All experiments are performed on the SQuAD dataset (Rajpurkar et al., 2016), where complete paragraphs are taken as input instead of just one or two sentences. We reformat the SQuAD dataset such that during training time, each source instance is a (paragraph, question) pair annotated with the gold answers, and the target is a question. Following the exact setup from MPGSN (Zhao et al., 2018), we split the SQuAD train set into train and validation set containing 77,526 and 9,995 instances respectively, and take the separate SQuAD dev set containing 10,556 instances as our test set.

### 4 Results and Analysis

Table 1 summarizes results of the automatic evaluation of the test set. As can be seen, our model significantly outperforms the state-of-the-art MPGSN on all metrics. The improvements on BLEU are especially substantial, the BLEU-4 score of MPGSN is 16.38, and ours (with copy incorporated) is 22.62, an improvement of 6.24, or 38%. This large performance difference demonstrates the effectiveness of our dynamic dictionary.

In Table 2 we present human evaluation results. We evaluate the quality of questions generated in terms of *syntactic* correctness, *semantic* correctness and *relevance* to the paragraph. The evaluation is performed on a randomly selected subset of 100 sentences from the test set. Each of the three evaluators are presented the 100 paragraph-question pairs for two variants of our model (with and without copy) and asked for a binary responses for all three

parameters. We averaged responses received by all three evaluators to compute the final scores. As can be seen, the incorporation of the copy attention improves performance, especially on relevance. We also measure the inter-rater agreement using Randolph’s free-marginal multirater kappa (Randolph, 2005). It can be observed that our quality metrics for both our models are rated as *substantial agreement* (Viera et al., 2005).

To explain how our model attends to different words in the source paragraph we visualize attention weights in Figure 3, which shows attention weights between question 2 generated by our model and the corresponding paragraph in Figure 1. We observe that the attention weight is high for words near the answer and the model attends to all relevant context rather than just the sentence containing the answer.

Model	Syntax		Semantics		Relevance	
	Score	Kappa	Score	Kappa	Score	Kappa
NQG <sub>dd</sub> [w/o copy]	89	0.68	<b>83</b>	0.69	43	0.67
NQG <sub>dd</sub> [with copy]	<b>94</b>	0.64	82	0.68	<b>71</b>	0.73

Table 2: Human evaluation results (columns “Score”) as well as inter-rater agreement (columns “Kappa”) for each of our two models on 100 questions from the test set. The scores are between 0 (worst) and 100 (best). Best results for each metric (column) are in **bold**.

We also note that our training is faster atleast by a factor of 2. We expected this since we replace a slightly expensive self-attention mechanism in the decoder of (Zhao et al., 2018) with a simpler dynamic dictionary and reusable copy attention.

### 5 Conclusion

Paragraph-level question generation (QG) is an important but challenging problem, mainly due to the challenge in effectively handling a longer context. We present a simple yet effective approach for automatic question generation from paragraphs. Besides using a standard global source dictionary, our RNN-based model incorporates a dynamic, paragraph-specific dictionary, and learns to switch between copying from the combined

vocabulary and generating a new word. Through our experiments, we demonstrate how our model outperforms the current state-of-the-art model in paragraph-level QG by a wide margin, for example by 6.24 BLEU-4 points, a 38% improvement.

## References

- Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. 2015. Microsoft COCO captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*.
- Xinya Du and Claire Cardie. 2018. Harvesting paragraph-level question-answer pairs from Wikipedia. In *ACL (1)*, pages 1907–1917.
- Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1342–1352.
- Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1631–1640.
- Vishwajeet Kumar, Kireeti Boorla, Yogesh Meena, Ganesh Ramakrishnan, and Yuan-Fang Li. 2018a. Automating reading comprehension by generating question and answer pairs. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 335–348. Springer.
- Vishwajeet Kumar, Ganesh Ramakrishnan, and Yuan-Fang Li. 2018b. A framework for automatic question generation from text using deep reinforcement learning. *arXiv preprint arXiv:1808.04961*.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.
- Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.
- Justus J Randolph. 2005. Free-marginal multirater kappa (multirater k [free]): An alternative to fleiss' fixed-marginal multirater kappa. *Online submission*.
- Linfeng Song, Zhiguo Wang, Wael Hamza, Yue Zhang, and Daniel Gildea. 2018. Leveraging context information for natural question generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, volume 2, pages 569–574.
- Sandeep Subramanian, Tong Wang, Xingdi Yuan, Saizheng Zhang, Adam Trischler, and Yoshua Bengio. 2018. Neural models for key phrase extraction and question generation. In *Proceedings of the Workshop on Machine Reading for Question Answering*, pages 78–88.
- Duyu Tang, Nan Duan, Tao Qin, Zhao Yan, and Ming Zhou. 2017. Question answering and question generation as dual tasks. *arXiv preprint arXiv:1706.02027*.
- Anthony J Viera, Joanne M Garrett, et al. 2005. Understanding interobserver agreement: the kappa statistic. *Fam Med*, 37(5):360–363.
- Tong Wang, Xingdi Yuan, and Adam Trischler. 2017. A joint model for question answering and question generation. *arXiv preprint arXiv:1706.01450*.
- Yao Zhao, Xiaochuan Ni, Yuanyuan Ding, and Qifa Ke. 2018. Paragraph-level neural question generation with maxout pointer and gated self-attention networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3901–3910.

# From Hero to Zéroe: A Benchmark of Low-Level Adversarial Attacks

Steffen Eger and Yannik Benz

Computer Science Department

Technische Universität Darmstadt, Germany

eger@aiphes.tu-darmstadt.de, yannik.benz@stud.tu-darmstadt.de

## Abstract

Adversarial attacks are label-preserving modifications to inputs of machine learning classifiers designed to fool machines but not humans. Natural Language Processing (NLP) has mostly focused on high-level attack scenarios such as paraphrasing input texts. We argue that these are less realistic in typical application scenarios such as in social media, and instead focus on *low-level* attacks on the character-level. Guided by human cognitive abilities and human robustness, we propose the first large-scale catalogue and benchmark of low-level adversarial attacks, which we dub *Zéroe*, encompassing nine different attack modes including visual and phonetic adversaries. We show that *RoBERTa*, NLP’s current workhorse, fails on our attacks. Our dataset provides a benchmark for testing robustness of future more human-like NLP models.

## 1 Introduction

Adversarial examples are label-preserving modifications to inputs of machine learning architectures. Their typical characteristic is that they cause little damage to humans but may maximally affect classifier performance, exposing their weaknesses and outlining the differences between human and machine text processing (Szegedy et al., 2014; Goodfellow et al., 2014; Eger et al., 2019).

While in computer vision, pixel-level attacks, which go unnoticed by humans, may lead to catastrophic failure, attacks in NLP are more challenging. Some attacks in NLP replace individual words by synonyms or hyponyms (Alzantot et al., 2018) or paraphrase whole sentences (Ribeiro et al., 2018). However, such *high-level* attacks are not only more difficult to compute (requiring available resources such as dictionaries or word embeddings) but they are also implausible in real-world scenarios such as spamming or posting in *social media*, as

users would need to know the training data and/or the inner workings of the machine learning models in order to identify candidate substitutions (or have unrestrained access to model predictions). In contrast, such users would typically use *low-level* attacks on characters, such as inserting placeholder symbols (e.g., underscores), mistyping words (e.g., *Hilter* for *Hitler*), or using phonetically similar sounding words (Tagg, 2011) to fool online detection models. To identify plausible such attack scenarios, human perceptual abilities play a decisive role. For instance, humans are guided by their senses, making them robust to, e.g., visual and phonetic attacks. Other scenarios to which humans have been shown robust include the removal of vowels from words or the shuffling of characters while keeping the initial and final letters fixed (see Section 2). However, the varieties in which text can be perturbed is certainly far from infinite, as (ordinary) humans, with all their cognitive constraints, still need to be able to decipher the text messages.

In this work, we provide the first large-scale catalogue for *low-level* (orthographic) attack scenarios. Our search is motivated by insights into human cognitive limitations and constraints and encompasses nine different attack modes (some of which are overlapping); cf. Table 1. We then examine the robustness of *RoBERTa* (Liu et al., 2019) to our attacks, finding that its performance can sometimes be severely decreased for our selection of attackers (up to the random guessing baseline); hence we call our benchmark *Zéroe*. The reason may be that our noises are not always natural, in the sense of having high support in large datasets such as CommonCrawl or Wikipedia, but they are still within the limits of cognitive abilities of ordinary humans. Finally, we show that under realistic conditions, standard adversarial training can restore

Attacker	Sentence
inner-shuffle	Aadreavsil aactks are hmarsels.
full-shuffle	idaAasvrler tstkaac are harmless.
intrude	A d v e r s a r i a l a t : t a : c k : s a r h } a r } m } less .
disemvowel	dvrsl ttcks r hrmlss .
truncate	Adversaria attack are harmles .
segment	Adversarial attacksare harmless .
typo	Adversrial attaxks are harmless .
natural noise	Adversarial attacs rae harmless .
phonetic	Advorcariel attaks are harmless .
visual	Ädužrsarīał attacks āre härūqlēš .

Table 1: Ten different modifications of the sentence “Adversarial attacks are harmless.”

*RoBERTa*’s performance only to a limited degree.<sup>1</sup>

## 2 Related Work

We classify adversarial attacks into *high-* and *low-level* attacks.<sup>2</sup>

**Attack Scenarios.** There are a variety of works that introduce **low-level** orthographic attacks.<sup>3</sup> **Ebrahimi et al. (2017)** trick a character-level neural text classification model by flipping the characters which cause most damage. Their approach is white-box, i.e., assumes access to the attack model’s parameters. **Eger et al. (2019)** exchange characters with similar looking ones and show that humans are robust to such visual perturbations, while machines may suffer severe performance drops. **Belinkov and Bisk (2017)** exchange adjacent letters on the keyboard with each other (keyboard typos) and introduce natural noise based on human typing errors extracted from different Wikipedia edit histories, as well as letter swaps. They use this natural and

<sup>1</sup>Code and data are provided at <https://github.com/yannikbenz/zeroe>.

<sup>2</sup>As one reviewer points out, a conceptual difference between high- and low-level attacks is that low-level attacks (as we define them) oftentimes induce linguistically corrupt text which can still be understood by humans, while high-level attacks operate in a noise-free environment to show the brittleness of systems even under ‘normal’ circumstances.

<sup>3</sup>Low-level adversarial attacks are in part examined by approaches to handle noisy user-generated text (**Baldwin et al., 2015**), with one difference being that attacks are often malicious in nature and may thus come in different forms.

synthetic noise to show the brittleness of machine translation (MT) systems, which contrasts with corresponding human robustness. **Ebrahimi et al. (2018)** also fool MT systems with character-level modifications. **Tan et al. (2020)** attack words by replacing them with morphological variants, which also mostly results in orthographic attacks (in English).

**High-level** attacks require a deeper understanding of the meaning and the syntactical structure of the sentence. **Jin et al. (2019)** generate semantically similar and syntactically correct adversarial examples by replacing words with suitable synonyms. **Hosseini et al. (2017)** and **Rodriguez and Rojas-Galeano (2018)** attack toxic detection systems by obfuscation, i.e., misspelling of the abusive words (a low-level attack), and via polarization, i.e., inverting the meaning of the sentences by inserting the word “not”. **Alzantot et al. (2018)** introduce an optimization-based algorithm to generate adversarial examples by replacing words in the input. Their generated words are semantically similar because they are nearest neighbors in the GloVe embedding space. They are also syntactically correct because they need to fit into the surrounding context with respect to the 1 billion words language model. **Iyyer et al. (2018)** generate syntactically correct paraphrases for a sentence. **Ribeiro et al. (2018)** use MT backtranslation to produce meaning-preserving adversaries. They generate adversarial examples for machine comprehension, sentiment analysis and visual question answering to show robustness issues in state-of-the-art models for each task. **Jia and Liang (2017)** insert semantically correct but irrelevant paragraphs into texts to fool neural reading comprehension models.

**Robustness.** **Adversarial training** is a commonly used technique to address adversarial attacks (**Szegedy et al., 2014**). The term may refer to calculating model gradients with respect to the input and inserting new training examples based on this gradient (**Goodfellow et al., 2014**). Alternatively, adversaries obtained from the attacker are inserted at train time (**Belinkov and Bisk, 2017; Alzantot et al., 2018; Eger et al., 2019**).

## 3 Catalogue of Attacks

We propose a catalogue of ten different attacks. Our intention is to suggest a maximally inclusive list of potential attacks under the constraint that humans are robust to them.

### 3.1 Attack protocol

Our attack protocol is *black-box* and *non-targeted* (Xu et al., 2019): we do not assume access to model parameters and our goal is to fool the system without any desired outcome in mind—in contrast, a spammer would want spam emails to be misclassified as non-spam, but not necessarily the reverse.

We parameterize attack levels by a *perturbation probability*  $p \in [0, 1]$ . With  $p$ , our goal is to attack  $p \cdot 100\%$  of all tokens in each sample in our dataset. To do so, for each sample  $w = (x_1, \dots, x_n)$ , we randomly and without replacement draw a token index  $i$  to perturb. We independently flip a coin with tail probability  $p$  to determine whether the token  $x_i$  should be attacked. We do so until either  $p \cdot 100\%$  of all tokens in  $w$  are perturbed or else if there are no more indices left.

### 3.2 Attacks

Some of our attacks, each of which operates on the character-level of an attacked word, are parametrized by a character-level perturbation probability  $\phi$ . For simplicity, we set  $\phi = p$  throughout, where  $p$  is the above defined word level perturbation probability.

**Inner Shuffle.** This randomly shuffles all letters in a word except for the first and last. This attack builds on the human ability to still comprehend words if the first and last letter remain intact (Rayner et al., 2006). We only allow change in words with length  $\geq 3$ .

**Full Shuffle.** This is the extreme case of the inner-shuffle perturbation where the constraint relating to initial and final letters is dropped. We include this attack for completeness, even though we do not assume high degrees of human robustness to it. We apply this to all words with length  $\geq 2$ .

**Intruders.** Inserting unobtrusive symbols (Hosseini et al., 2017) in words is a typical phenomenon in social media, e.g., to avoid censorship. Depending on the symbols chosen, an attack may have little effect on humans. We choose the inserted symbol randomly but in case of multiple insertions into one word keep the symbol identical. We allow the following symbols to be inserted: !"#\$%&'()\*+,-./;:<=>?@[\]^`{|}, including whitespace. The perturbation probability  $\phi$  additionally influences the number of insertions taking place. For each two characters,  $\phi$  indicates how

likely the insertion of a symbol between them is. We apply this attack to all words with length  $\geq 3$ .

**Disemvoweling.** This removes all vowels (a, e, i, o, u) from a word. If a word only consists of vowels, it will be ignored to prevent it from being deleted. Words with length  $\leq 3$  are skipped to maintain readability. Disemvoweling is a common feature of SMS language and on social media presumed to require little cognitive effort for humans (Boyd et al., 2010).

**Truncating.** This removes a fixed number of letters from the back of a word. We only cut the last letter from words of length  $\geq 3$  to maintain readability. Predicting word endings from beginnings is considered an easy task for humans (Elman, 1995).

**Segmentation.** This joins multiple words together into one word. Here, the perturbation level is the probability to merge the first two adjacent words. Each following word gets a lower probability to get merged ( $\phi^2, \dots, \phi^n$ ) to prevent ‘giant’ words. We do not apply this attack to sequence tagging tasks such as POS, because the joined words would have no proper tag, making evaluation more difficult. The ability of humans to segment unsegmented input is already acquired during infancy (Goldwater et al., 2009).

**Keyboard Typos.** We adopt this attack from Be-linkov and Bisk (2017) and adapt it to our workflow. Hereby, adjacent letters on the English keyboard are replaced by each other randomly. This simulates human typing errors. The higher the perturbation probability  $\phi$ , the more characters are exchanged by adjacent letters.

**Natural Typos.** Words are replaced by natural human errors from the Wikipedia edit history (Be-linkov and Bisk, 2017) which contains multiple sources of error: phonetic errors, omissions, morphological errors, key-swap errors and combinations of them.

**Phonetic.** An ideal phonetic attack leaves the pronunciation of a word intact but alters its spelling. Phonetic attacks are common especially in English with its irregular mapping of pronunciation and spelling. They do not only occur as mistakes but also as a form of creative language use (Tagg, 2011).

**Visual.** Visual attacks are based on the idea that humans may easily recognize similar looking sym-

bols (Eger et al., 2019). We replace each character in the input sequence with one of its 20 visual nearest neighbors in the visual space defined below. This attack is also parameterized by  $\phi$ : we replace each letter in a word i.i.d. randomly with probability  $\phi$ .

We observe that our attacks are *not* directly comparable. For example, at some perturbation level  $p$ , truncate removes  $O(p \cdot n)$  characters, where  $n$  is sentence length. In contrast, intruders inserts  $O(p^2 \cdot n \cdot m)$  characters, where  $m$  is a bound on word length.

### 3.3 Implementation of Visual and Phonetic Attacks

We describe details of phonetic and visual attacks below, as they are more involved.

**Phonetic Embeddings and Attacks.** In order to replace words by phonetically similar ones, we use two stages. First, we train two Seq2Seq models to translate a letter string into its phonetic representation and vice versa. We use the Combilex dataset to do so (Richmond et al., 2010). In addition to that, we induce *phonetic word representations*, i.e., a vector space where two words are close if they are pronounced alike. We use an InferSent-like architecture to do so (Conneau et al., 2017). Details are given in the appendix. When a word  $x$  should be phonetically perturbed, we run the first Seq2Seq model to obtain a phonemic representation and then convert this back to a letter string  $\tilde{x}$  (as in backtranslation in MT). We finally keep  $\tilde{x}$  when it is phonetically similar to  $x$ . We added the latter step because we observed that some resulting words  $\tilde{x}$  had very different pronunciation than  $x$  after the backtranslation.

**Visual embeddings.** In order to generate visual character embeddings, we used an architecture introduced by Larsen et al. (2016) as a combination of GAN and VAE, called VAEGAN. The model is able to learn embeddings which encode high-level abstract features. This property is desirable in our case, because humans rely on abstract features (Dehaene and Cohen, 2011), i.e., shape and spatial relation of the letter, instead of pixels while reading. The model is described in the appendix.

To obtain visual character embeddings, we generate a grayscale image of size  $24 \times 24$  for each character in the Basic Multilingual Plane (BMP; 65k characters) of the standard Unicode character set with *Pillow*. The VAEGAN is trained on

the full BMP dataset. After training, we compute 256-dimensional visual letter embeddings by encoding the respective letter image with the encoder of the VAEGAN. The quality of the embeddings can be derived via the models’ ability to properly reconstruct an image from them, see Figure 7 in the appendix.

## 4 Experimental Setup

### 4.1 Base model and datasets

Our base architecture used in all experiments is *RoBERTa* (Liu et al., 2019). RoBERTa is a robustly optimized extension of BERT that has been trained (i) for longer, (ii) on more data, and (iii) without the next sentence prediction task. RoBERTa has been shown to outperform BERT on a variety of benchmark tasks, including those contained in GLUE (Wang et al., 2018). We study the performance of *RoBERTa* in our attack scenarios on three different NLP tasks. Dataset statistics are shown in Table 2.

**POS tagging** is a sequence tagging task where each token in the input needs to be labeled with its respective POS tag. We use the English universal dependency dataset with 17 different tags (Nivre et al., 2016).

**NLI** is a classification task in which the relation of a sentence pair must be predicted. Relation labels are *neutral*, *contradiction* and *entailment*. We use SNLI (Bowman et al., 2015).

**Toxic Comment Classification (TC)** labels sentences (typically from social media platforms) with one or several toxicity classes. Possible labels are: *toxic*, *obscene*, *threat*, *insult* and *identity hate*. For this task, we choose the jigsaw toxic comment challenge dataset from kaggle<sup>4</sup>. The current best performance on the leaderboard has an AUCROC (area under the receiver operations characteristic curve) score of 98.8%.

### 4.2 Results

We consider the cases of *low* ( $p = 0.2$ ), *mid* ( $p = 0.5$ ) and *high* ( $p = 0.8$ ) attack levels.

In Figure 1, we plot the performance of *RoBERTa* for the three tasks POS, NLI and TC individually as we perturb the test data using our attackers. Detailed numbers are reported in Table 6

<sup>4</sup><https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge>

Task	Dataset	Train	Test	Clean score
<b>POS Tagging</b>	Universal Dependencies (part)	13k	2k	96.95
<b>NLI</b>	Stanford Natural Language Inference	550k	10k	90.41
<b>Multilabel Classification</b>	Toxic Comment	560k	234k	0.93

Table 2: Overview of the NLP tasks used in this work. Clean scores are scores from training and testing on clean data.

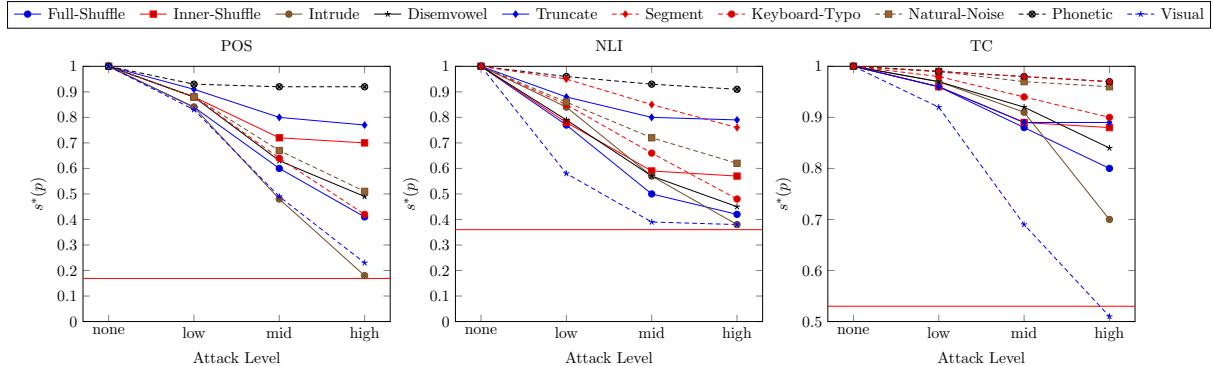


Figure 1: Performance decreases of RoBERTa on the three downstream tasks: POS, SNLI and TC. Red lines indicate the random guessing baseline.

in the appendix. We report scores relative to the model performances on the clean test set:

$$s^*(p) = \frac{s(p)}{s(0)}, \quad p \in \{0, 0.2, 0.5, 0.8\} \quad (1)$$

where  $s(0)$  is the task specific performance on clean data listed in Table 2 and  $s(p)$  is the performance for attack level  $p$ . Scores are measured in accuracy for POS and NLI, and in AUCROC for TC classification. Clean performance scores depend on the specific task and dataset. For example, NLI has a worst score of around 33% accuracy (majority label) and POS has a corresponding worst score of around 16% accuracy. The worst performance of TC is reached at AUCROC score of 50%—at this point, the model is no longer able to distinguish between the different classes. We mark these values relative to the tasks’ best performance ( $s(0)$ ) in Figure 1 as red lines.

Each task suffers performance decreases from each attacker. The higher the perturbation level, the lower the model performance.

The **phonetic** attack is the least effective for all tasks with maximally 10 percentage points (pp) performance decrease with the *highest* perturbation probability of 0.8. The **truncate** attack yields higher performances decreases in all three tasks,

being roughly twice as effective. The performance decreases by 10pp from *none* to *low* and additional 10pp from *low* to *mid*. Increasing the attack level beyond that does not cause further harm, especially for NLI and TC. Concerning the **segmentation** attack, for NLI, it leads to a similar performance decrease as the truncate attack for *small p*, but becomes more successful as the perturbation level increases to *mid* and *high*. For TC, the performance decrease is almost identical to the phonetic attack.

We notice a linear decrease in performance for each task when increasing the perturbation level of the **natural-noise** attack. Especially POS and NLI suffer a strong performance deterioration of around 40pp and 50pp for the *highest* attack level. Both lose 15pp to 20pp performance per attack level increase.

**Full-** and **inner-shuffle** randomize the order in an input word but humans are more robust to inner-shuffle. Full-shuffle also affects *RoBERTa* more than inner-shuffle. It tends to be one of the strongest attack scenarios, while inner-shuffle typically ranks in the midfield.

The **disemvowel** attack has different effects in different tasks. For POS, it is almost identical to the natural-noise attack with a slightly stronger impact of 5pp for *mid* and 3pp for *high* and a maxi-

mum on 50pp. NLI loses around 20pp performance on *low* and it decreases an additional 20pp by increasing the level to *mid*, and reaches its greatest decrease by 55pp on *high*. In TC, model performances decrease linearly from *none* to *low* and *mid* by 8pp each. The *high* attack level doubles to a total of 15pp performance loss. The **keyboard-typo** attacks have median impact throughout tasks and attack levels.

The **intrude** attack is among the most severe attacks across all three tasks. For TC, the *low* and *mid* attack levels have a relatively low impact compared to *high* which yields a performance loss of 30pp. It decreases model performance the most on the POS task by above 80pp. Especially for both sentence-based tasks NLI and TC, the **visual** attack decreases are also among the most severe, while *RoBERTa* is marginally more robust on the POS task. Even for the *low* perturbation level, the NLI model suffers from more than 40pp performance decrease. The performance for high  $p$  even falls below the red line marked as our lower bound baseline.

### 4.3 Defenses

In the following, we report the performance increase from shielding the methods with adversarial training:

$$\Delta_\tau(p) := \frac{\sigma(p)}{s(0)} - s^*(p) \quad (2)$$

where  $\sigma(p)$  is the score for each task with one of two defense methods  $\tau$ :

- **1-1 adversarial training**( $\alpha, \beta$ ): Here, we train on a mixture of *low*, *mid*, *high* attacked data (each perturbation level is roughly equally likely to appear in the training data). We attack with some attacker  $\alpha$  and measure performance when the test data is attacked with attacker  $\beta$ .
- **leave-one-out** (LOO): Here, we train on a mix of all attackers except for the one with which the test data is attacked. The train data contains an equal mix of data from each attacker and attack level.

#### 4.3.1 Adversarial Training

**1-1** ( $\alpha, \alpha$ ) In Figure 2, we report the performance of our models each trained on perturbed data and evaluated against *the same kind* of perturbation. This gives an unrealistic upper bound since the defender would have to know how it is being attacked.

For POS, the adversarially trained models lose a bit of their performance on clean data, but their performance on perturbed data improves, especially against intrude and truncate for the *low* attack level. The robustness improvements for the remaining attackers are very similar and range from 3pp increase for the natural-noise attack to 8pp for the disemvowel attack. With one exception, the improvement at large perturbation levels  $p$  is highest, and obtains a maximum improvement of 40pp for inner-shuffle.

For NLI, the models again tend perform worse on clean data. As the perturbation level increases, we see a smooth and steady increase of the values  $\Delta_\tau(p)$  across all attackers. Improvement is best for intrude which was also among the most damaging attacks.

For TC, model performances increase also on clean data, which is likely due to the nature of the task. As the attack level increases,  $\Delta_\tau(p)$  gradually further increases across tasks. For *high*, largest increase is again observed for intrude as well as for visual, which also had largest impact in the non-shielded setting.

**1-1** ( $\alpha, \beta$ ) In Figure 3, we show all 1-1 values for different combination of attackers on train ( $\alpha$ ) and test data ( $\beta$ ). We see that the diagonal ( $\alpha = \beta$ ) always profits considerably, but the off-diagonal can be positive or negative, depending on the choice of  $\alpha$  and  $\beta$ . We clearly see that (1) truncate, disemvowel, keyboard-typo, natural noise, visual, and intruders are similar in the sense that training on them shields against their attacks at test time. (2) Full-shuffle and inner-shuffle form a second group and (3) phonetic attacks a third group. This is to some degree a natural clustering, as (1) removes or replaces characters, (2) destroys the order of words, and (3) modifies entire words using more complex operations. visual is an outlier in group (1), since it improves no matter what attacks are added at train time.

**Leave-One-Out** Figure 4 shows the performance of our models when trained on a mixture of all attackers except the one evaluated on. This is the most plausible scenario of model defense in the case of an unknown new attack scenario at test time.

For POS, the performance against the phonetic attack remains mostly unchanged, while  $\Delta_\tau(p)$  increases as a function of  $p$  against natural-noise,

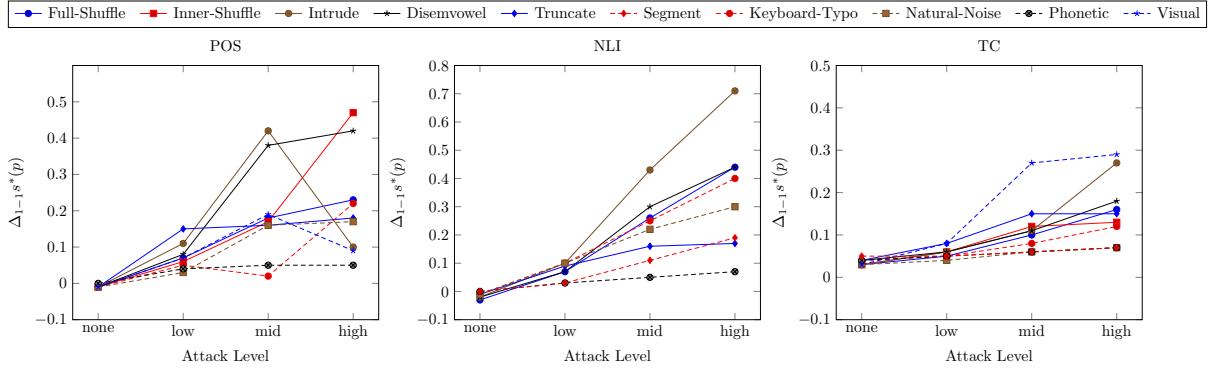


Figure 2: Performance improvements of the models adversarial trained and evaluated individually on the attacker introduced in Section 3 for POS left, NLI mid and TC right. Performance measured in  $\Delta_\tau(p)$  defined in Eq. 2.

		FS	IS	INT	DIS	TRUN	KEY	NAT	PH	VIS
FS	low	6.35	-0.46	-2.13	0.49	-0.29	-0.69	-1.21	0.02	0.96
	high	17.59	-0.1	-4.1	4.03	1.81	-0.45	1.48	-1.23	2.24
	mid	21.8	0.48	-4.36	7.59	4.37	0.11	3.93	-1.68	1.97
IS	low	2.01	6.02	0.1	0.72	-0.31	1.13	-0.33	0.14	1.74
	mid	3.7	16.71	0.89	4.24	2.4	3.53	0.41	0.08	4.08
	high	3.77	18.29	1.25	5.25	2.52	3.89	0.59	-0.28	4.78
INT	low	-2.19	-1.04	11.3	0.4	0.63	5.46	0.85	-2.16	5.3
	mid	-7.79	-4.89	41.5	10.77	7.26	12.38	6.71	-2.54	13.93
	high	-7.69	-6.22	62.81	9.8	6.59	11.4	29.56	-2.27	4.3
DIS	low	-2.95	-2.79	-0.56	8.07	0.67	0.19	0.48	-0.97	0.35
	mid	-4.75	-1.42	1.44	27.73	6.05	5.18	5.68	-1.17	3.27
	high	-4.39	0.42	2.76	41.44	9.2	8.68	9.39	-1.82	4.62
TRUN	low	-0.4	-1.07	-0.23	0.4	6.01	0.88	1.46	0.06	0.23
	mid	-0.22	-1.38	0.93	2.18	15.87	3.18	4.93	-0.61	1.58
	high	-0.2	-1.27	1.17	2.79	17.88	3.53	5.73	-0.78	1.72
KEY	low	-1.36	-1.9	0.11	0.3	0.24	5.05	0.91	-0.95	1.93
	mid	-1.89	-1.42	1.4	2.82	1.66	11.04	4.65	-1.79	4.12
	high	-4.07	-2.49	3.96	3.26	4.65	22	6.72	-3.27	5.94
NAT	low	-1.42	-2.77	-0.13	-0.21	-0.6	0.49	3.18	-0.93	1.14
	mid	-0.31	-2.86	0.47	2.77	1.03	2.61	15.62	-0.77	2.15
	high	-2.23	-3.05	2.33	3.46	4.04	4.2	16.7	-1.27	3.19
PH	low	-1.8	-1.96	-1.73	-1.22	-1.41	-1.76	-0.81	4.27	-1.15
	mid	-2.41	-2.18	-1.97	-1.45	-1.73	-2.16	-1.02	5.3	-1.42
	high	-2.21	-2.16	-1.95	-1.44	-1.69	-2.1	-0.95	5.41	-1.39
VIS	low	1.55	1.61	2.58	2.12	2.09	3.69	1.74	0.68	7.43
	mid	5.01	6.65	7.91	9.13	7.41	9.92	5.09	1.18	18.44
	high	1.46	3.62	-0.25	7.87	6.6	7.99	3.81	1.22	8.94

Figure 3: 1-1 ( $\alpha, \beta$ ) adversarial training for POS. Column: train, row: test. Numbers give values  $\Delta_\tau(p)$ , see Eq. (2). Red colors give performance decreases, relative to the results on clean data; blue colors show increases.

inner-shuffle, full-shuffle, truncate and keyboard-typo. The best defense is against natural-noise with 3pp for *low* and 7pp for *mid* and *high*. Shielding against visual, intrude and disemvowel attacks yields lower values  $\Delta_\tau(p)$  on attack level *high* compared to *mid*. Overall, we see mild improvements compared to the unshielded situation, but expectedly, these are lower than for 1-1 shielding.

For NLI, the performance against keyboard-typo, full-shuffle, inner-shuffle, natural-noise and truncate exhibits steady improvements with increasing attack level which range from 10pp to 20pp for attack level *mid* and *high*. The performances against intrude and disemvowel also show steady improvements with the attack levels but are generally higher with up to 29pp. For attack level *low*, the perfor-

mance improvement against the visual attacker is with 20pp more than twice the value of the others. This improvement diminishes in the *mid* and *high* attack levels and even drops there below the improvements against most of the other attackers.

In the TC task, the performance against visual improves even for *low* level to 8pp, increases for *mid* to 23pp and maximizes to 29pp total improvement for attack level *high*. The performance against the intrude attack is also very good: for *low* attack level the improvement (11pp) is even higher compared to visual (8pp). The performances against full-shuffle, inner-shuffle, disemvowel, segment, keyboard-typo, natural-noise and phonetic behave similar for attack level *low* and *mid* with 4pp to 7pp total improvement. Shielding against full-swap and

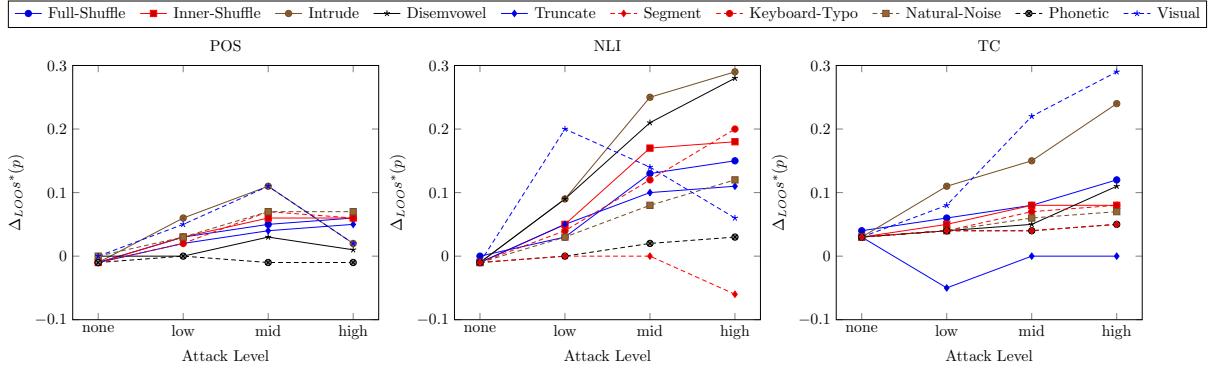


Figure 4: Leave-one-out defense: Performance improvements of the models adversarial trained on all attackers introduced in Section 3 except the one they are evaluated on for POS left, NLI mid and TC right. Performance measured in  $\Delta_\tau(p)$  defined in equation 2.

disemvowel is slightly better than the last group. There is no overall positive effect for truncate.

#### 4.4 Discussion

Overall, the phonetic attack was least effective. We assume this is because few words were changed overall as a considerable amount of phonetic replacements were either identical to the input and some were even discarded.

The truncate attack performed better than the phonetic attack in all three tasks but it still remained low overall, possibly as we truncated only by 1 character, leading to small changes in the appearance of a word.

We attribute the low impact of the segmentation attack to *RoBERTa*'s BPE encoding, which apparently allows it to partly de-segment unsegmented input. We observe that some attacks (e.g., segmentation, keyboard-typo, and natural-noise) have less effect in TC compared to POS and NLI, possibly because of higher natural occurrences of these phenomena in the TC dataset.

The intrude and visual attacks are among the strongest. This is *not only* because they are doubly parametrized unlike many others—i.e., for *high* attacks, not only the majority of words is attacked but also the majority of characters within a word—since they are also effective at *low* attack levels. We partly attribute their success to the fact that they cause a high out-of-vocabulary rate for *RoBERTa* and tend to increase the number of input tokens, as they cause *RoBERTa* to segment the input at unknown characters. This may lead to the number of input tokens exceeding *RoBERTa*'s built-in max token size, leading to cutting off the ending of the sentence.

Rank	POS	NLI	TC
1	1-1 ( $\alpha, \alpha$ )	16	1-1 20
2	LOO	4	LOO 10
3	1-1 ( $\alpha, \beta$ )	1	1-1 3

Table 3: Different defense approaches ranked by the average robustness improvement over all attackers. Improvement in percentage points (pp; rounded).

In Table 5 (appendix), attacks are ranked (for *high* attack level) by the performance degradation caused to the model for each individual task. In line with our previous discussion, the visual and the intrude attackers are always the both best performing, followed by full-shuffle (which we deemed as unrealistic as it would also destroy human perception abilities). Figure 5 shows the relationship between the amount of text perturbed in a test dataset and the performance deterioration a model suffers. This shows a clear (linear) trend and indicates that a successful attacker most importantly needs to attack many characters of a text to be effective, despite all individual qualitative differences between the attackers discussed above.

In Table 3, a ranking of defense strategies is given. 1-1 ( $\alpha, \alpha$ ) performs best, but is unrealistic. LOO is a robust alternative for unknown new attacks. The effectiveness of LOO as defense is also a further justification for designing multiple attack models.

## 5 Conclusion

We provided the first large-scale catalogue for low-level adversarial attacks, providing a new simple benchmark for testing real-world robustness of future deep learning models. We further showed

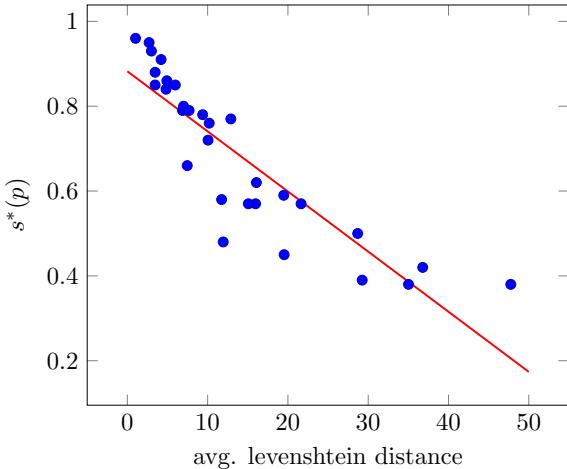


Figure 5: Relation between the amount of text perturbed (measured in edit distance) in a test data set and  $s^*(p)$ , the performance decrease a model suffers.

that one of the currently most successful deep learning paradigms, *RoBERTa*, is not robust to our benchmark, sometimes suffering catastrophic failure. While many of our errors could probably be addressed by placing a correction layer in front of *RoBERTa* (Choudhury et al., 2007; Pruthi et al., 2019), we believe that our findings shed further light on the differences between human and machine text processing, which deep models eventually will have to *innately* overcome for true AI to become a viable prospect.

## Acknowledgments

We thank the anonymous reviewers for their useful comments and suggestions. Steffen Eger has been funded by the HMWK (Hessisches Ministerium für Wissenschaft und Kunst) as part of structural location promotion for TU Darmstadt in the context of the Hessian excellence cluster initiative “Content Analytics for the Social Good” (CA-SG).

## References

- Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. *Generating Natural Language Adversarial Examples*. *arXiv e-prints*, page arXiv:1804.07998.
- Timothy Baldwin, Marie-Catherine de Marneffe, Bo Han, Young-Bum Kim, Alan Ritter, and Wei Xu. 2015. Shared tasks of the 2015 workshop on noisy user-generated text: Twitter lexical normalization and named entity recognition. In *Proceedings of the Workshop on Noisy User-generated Text*, pages 126–135.
- Yonatan Belinkov and Yonatan Bisk. 2017. *Synthetic and Natural Noise Both Break Neural Machine Translation*. *arXiv preprint arXiv:1711.02173*.
- Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. *A large annotated corpus for learning natural language inference*. In *Conference Proceedings - EMNLP 2015: Conference on Empirical Methods in Natural Language Processing*, pages 632–642.
- Danah Boyd, Scott Golder, and Gilad Lotan. 2010. *Tweet, tweet, retweet: Conversational aspects of retweeting on twitter*. In *Proceedings of the 2010 43rd Hawaii International Conference on System Sciences, HICSS ’10*, page 1–10, USA. IEEE Computer Society.
- Monojit Choudhury, Rahul Saraf, Vijit Jain, Animesh Mukherjee, Sudeshna Sarkar, and Anupam Basu. 2007. Investigation and modeling of the structure of texting language. *International Journal of Document Analysis and Recognition (IJDAR)*, 10(3-4):157–174.
- Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. 2017. *Supervised Learning of Universal Sentence Representations from Natural Language Inference Data*. *arXiv e-prints*, page arXiv:1705.02364.
- Stanislas Dehaene and Laurent Cohen. 2011. The unique role of the visual word form area in reading. *Trends in Cognitive Sciences*, 15(6):254 – 262.
- Javid Ebrahimi, Daniel Lowd, and Dejing Dou. 2018. *On Adversarial Examples for Character-Level Neural Machine Translation*. *arXiv e-prints*, page arXiv:1806.09030.
- Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2017. *HotFlip: White-Box Adversarial Examples for Text Classification*. *arXiv e-prints*, page arXiv:1712.06751.
- Steffen Eger, Gözde Gül Şahin, Andreas Rücklé, Jihung Lee, Claudia Schulz, Mohsen Mesgar, Krishnakant Swarnkar, Edwin Simpson, and Iryna Gurevych. 2019. *Text processing like humans do: Visually attacking and shielding NLP systems*. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1634–1647, Minneapolis, Minnesota. Association for Computational Linguistics.
- Jeffrey L Elman. 1995. Language as a dynamical system. *Mind as motion: Explorations in the dynamics of cognition*, pages 195–223.
- Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. 2009. *A bayesian framework for word segmentation: Exploring the effects of context*. *Cognition*, 112(1):21 – 54.

- Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. [Explaining and Harnessing Adversarial Examples](#). *arXiv e-prints*, page arXiv:1412.6572.
- Hossein Hosseini, Sreeram Kannan, Baosen Zhang, and Radha Poovendran. 2017. [Deceiving Google’s Perspective API Built for Detecting Toxic Comments](#). *arXiv e-prints*, page arXiv:1702.08138.
- Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. [Adversarial Example Generation with Syntactically Controlled Paraphrase Networks](#). *arXiv e-prints*, page arXiv:1804.06059.
- Robin Jia and Percy Liang. 2017. [Adversarial examples for evaluating reading comprehension systems](#). In *EMNLP 2017 - Conference on Empirical Methods in Natural Language Processing, Proceedings*, pages 2021–2031.
- Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2019. [Is BERT Really Robust? A Strong Baseline for Natural Language Attack on Text Classification and Entailment](#). *arXiv e-prints*, page arXiv:1907.11932.
- Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. 2016. [Autoencoding beyond pixels using a learned similarity metric](#). In *33rd International Conference on Machine Learning, ICML 2016*, volume 4, pages 2341–2349.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. [RoBERTa: A Robustly Optimized BERT Pretraining Approach](#). *arXiv e-prints*.
- Joakim Nivre, Marie Catherine De Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. [Universal dependencies v1: A multilingual treebank collection](#). In *Proceedings of the 10th International Conference on Language Resources and Evaluation, LREC 2016*, pages 1659–1666.
- Danish Pruthi, Bhuvan Dhingra, and Zachary C. Lipton. 2019. [Combating adversarial misspellings with robust word recognition](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5582–5591, Florence, Italy. Association for Computational Linguistics.
- Keith Rayner, Sarah J. White, Rebecca L. Johnson, and Simon P. Liversedge. 2006. [Raeding wrods with jubmled lettres: There is a cost](#). *Psychological Science*, 17(3):192–193.
- Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. [Semantically equivalent adversarial rules for debugging NLP models](#). In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 856–865, Melbourne, Australia. Association for Computational Linguistics.
- Korin Richmond, Robert Clark, and Sue Fitt. 2010. On generating combilex pronunciations via morphological analysis. In *Proceedings of the 11th Annual Conference of the International Speech Communication Association, INTERSPEECH 2010*, pages 1974–1977.
- Nestor Rodriguez and Sergio Rojas-Galeano. 2018. [Shielding Google’s language toxicity model against adversarial attacks](#). *arXiv e-prints*, page arXiv:1801.01828.
- Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2014. [Intriguing properties of neural networks](#). In *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*.
- Caroline Tagg. 2011. Wot did he say 01” could u not c him 4 dust?: Written and spoken creativity in text messaging. *Transforming literacies and language: Multimodality and literacy in the new media age*, page 223.
- Samson Tan, Shafiq Joty, Min-Yen Kan, and Richard Socher. 2020. [It’s morphin’ time! Combating linguistic discrimination with inflectional perturbations](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2920–2935, Online. Association for Computational Linguistics.
- Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. [GLUE: A multi-task benchmark and analysis platform for natural language understanding](#). In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.
- Han Xu, Yao Ma, Haochen Liu, Debayan Deb, Hui Liu, Jiliang Tang, and Anil K. Jain. 2019. [Adversarial Attacks and Defenses in Images, Graphs and Text: A Review](#). *arXiv e-prints*, page arXiv:1909.08072.

<b>Homophone Abbreviation</b>	byte I love you too!	bite I luv U 2!
-------------------------------	-------------------------	--------------------

Table 4: Example of a homophone and a typical “internet slang” abbreviation.

## A Appendices

### A.1 Phonetic and visual embeddings

**Phonetic Word Embeddings.** To induce phonetic word embeddings, we adopt the Siamese network of InferSent (Conneau et al., 2017). InferSent was originally designed to induce vector representations for two sentences from which their entailment relation was inferred. We adapt InferSent to encode two words so that their phonological similarity can be inferred: *identical*, *very similar*, *similar* and *different*. We use the BiLSTM max-pooling approach from the original InferSent paper, where we set the induced phonetic embeddings size to 100.

We build our own dataset for phonetic similarity by leveraging data from different sources. Initially, we use Combilex (Richmond et al., 2010), which gives phonetic representations for standard (American) English words. We calculate the normalized edit distance between the phonemes of each word pair to determine the phonetic similarity of two words:

$$\text{sim}_{\text{ph}}(\pi_1, \pi_2) = 1 - \frac{d(\pi_1, \pi_2)}{\min(|\pi_1|, |\pi_2|)} \quad (3)$$

where  $\pi_i$  are phonetic sequences for underlying words and  $d$  is the edit-distance. We then map the words into 4 different classes: *identical* ( $\text{sim}_{\text{ph}} = 0$ ), *very similar* ( $0 < \text{sim}_{\text{ph}} < 0.1$ ), *similar* ( $0.1 < \text{sim}_{\text{ph}} < 0.3$ ) and *different* ( $0.3 < \text{sim}_{\text{ph}}$ ). To keep the training data for each class more balanced, we added handcrafted and crawled samples, e.g., homophones. We also wanted to include “internet slang” style phonetic replacements like in Table 4. We therefore crawled them and added them to the bins *identical* and *very similar* based upon manual inspection. Overall, we compiled 5k examples for each of our four labels. The *similar* and *different* bins consist only of data from Combilex, whereas the *identical* and *very similar* bin contains 1.3k samples from Combilex and 3.7k crawled samples. References for crawled sites are given in A.2.

**Visual Embeddings.** The model reduces the dimension of input  $x$ , e.g., an image, by applying

multiple convolutional steps in the *encoder* to compute the latent representation  $z$  of  $x$ . Afterwards, it reconstructs the original input  $x$  in the *decoder* by applying multiple deconvolutional steps to  $z$ . This reconstructed version of  $x$  is called  $\tilde{x}$ . Additionally, a second input  $z_p$  sampled from  $\mathcal{N}(0, I)$  is inserted into the *generator* to obtain  $x_p$ . *Decoder* and *generator* perform the same task on different inputs; they can be considered as identical and therefore share their parameters. The *discriminator* takes  $x$ ,  $\tilde{x}$  and  $x_p$  as inputs and discriminates which input is a real training sample and which is a fake. Figure 6 illustrates the working of the architecture.

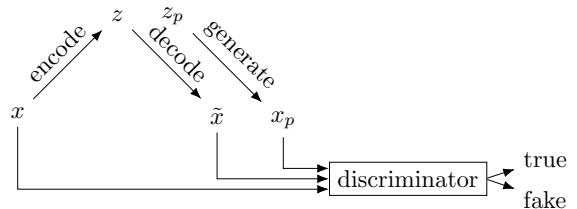


Figure 6: Schematic representation of Variational Autoencoder Generative Adversarial Network (VAEGAN) taken and adapted from Larsen et al. (2016).  $z$  can be decomposed as  $z = \mu + \sigma$  and is used to sample  $z_p = \mu + \sigma\epsilon$  where  $\epsilon$  is noise defined as  $\epsilon \sim \mathcal{N}(0, I)$



Figure 7: Reconstruction of images after being compressed to its latent representation and decompressed back to the original data distribution.

Figure 8 gives an impression of the encoded visual similarity.

### A.2 Homophone resources

List of used resources to gather homophones.

- <https://7esl.com/homonyms/>
- <https://www.englishclub.com/pronunciation/homophones-list.html>
- <https://www.thoughtco.com/homonyms-homophones-and-homographs-a-b-1692660>
- <http://www.singularis.ltd.uk/bifroest/misc/homophones-list.html>

Ranking	POS	NLI	TC
1	Intrude	Visual & Intrude	Visual
2	Visual	-	Intrude
3	Full-Shuffle	Full-Shuffle	Full-Shuffle
4	Keyboard-Typo	Disemvowel	Disemvowel
5	Disemvowel	Keyboard-Typo	Inner-Shuffle
6	Natural-Noise	Inner-Shuffle	Truncate
7	Inner-Shuffle	Natural-Noise	Keyboard-Typo
8	Truncate	Segment	Natural-Noise
9	Phonetic	Truncate	Segment
10	-	Phonetic	Phonetic

Table 5: Ranking on harmfulness of the attackers on POS, NLI, TC on attack level *high*.

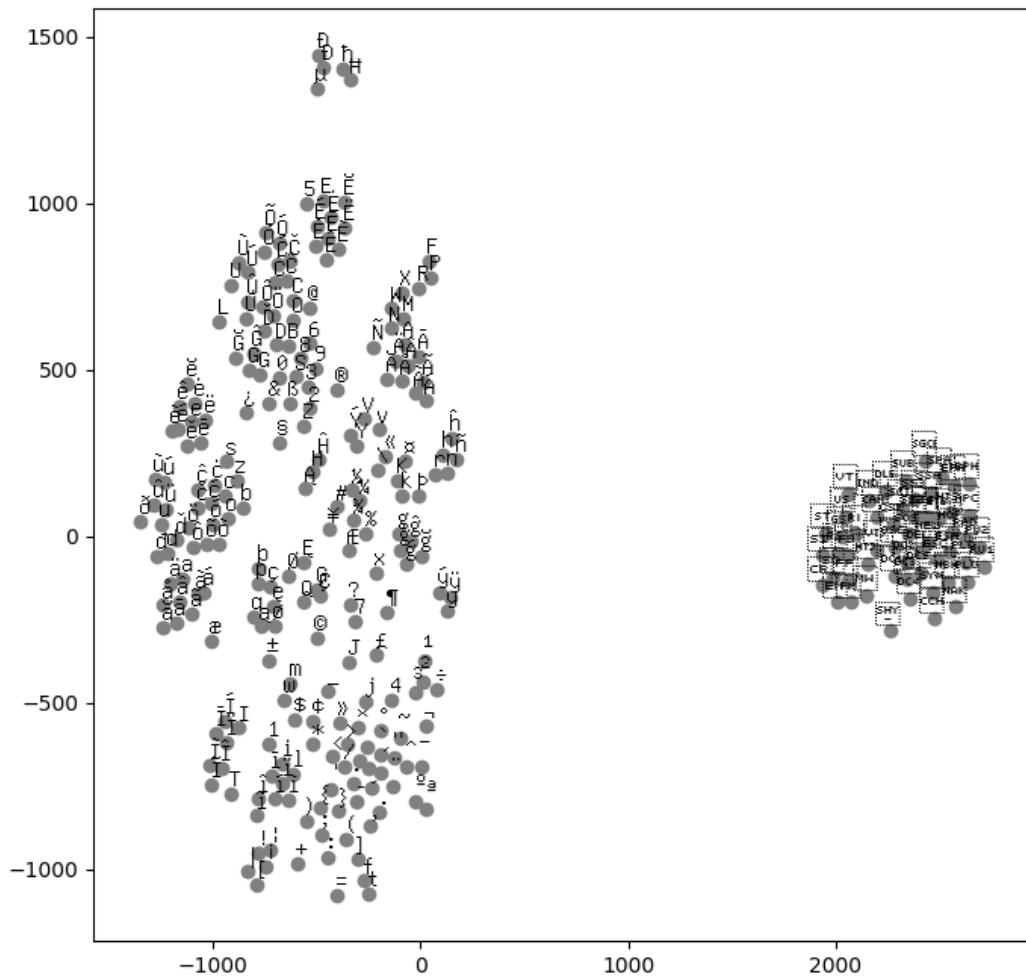


Figure 8: tSNE plot of our character embedding space. As can be seen similar looking characters are clustered.

- <https://web.archive.org/web/20160825095711/>
- <http://people.sc.fsu.edu/~jburkardt/fun/wordplay/multinyms.html>
- <http://homophonelist.com/homophones-list/>
- <https://web.archive.org/web/20160825095711/>
- <http://homophonelist.com/homophones-list/>
- [https://www.webopedia.com/quick\\_ref/textmessageabbreviations.asp](https://www.webopedia.com/quick_ref/textmessageabbreviations.asp)
- <https://www.smart-words.org/abbreviations/text.html>
- [https://en.wiktionary.org/wiki/Appendix:English\\_dialect-independent\\_homophones](https://en.wiktionary.org/wiki/Appendix:English_dialect-independent_homophones)
- [https://en.wiktionary.org/wiki/Appendix:English\\_dialect-dependent\\_homophones](https://en.wiktionary.org/wiki/Appendix:English_dialect-dependent_homophones)

### A.3 Detailed Result Tables

Hyperparameters of our models can be found in the github accompanying the publication (<https://github.com/yannikbenz/zeroe>). The following tables give detailed results of our experiments.

<b>Attack</b>	<b>Mode</b>	<b>Accuracy</b>		<b>AUCROC</b>
		<b>POS</b>	<b>NLI</b>	
None	-	96.65	90.41	0.93
Full-Swap	low	82.14	70.35	0.90
	mid	58.14	45.70	0.83
	high	40.47	38.35	0.74
Inner-Swap	low	85.96	67.70	0.90
	mid	70.53	53.35	0.83
	high	67.95	51.55	0.82
Intrude	low	81.42	75.97	0.91
	mid	46.91	52.25	0.85
	high	18.15	34.70	0.66
Disemvowel	low	85.24	72.24	0.91
	mid	61.50	51.62	0.86
	high	44.69	41.00	0.79
Truncate	low	88.57	79.83	0.90
	mid	77.40	72.87	0.84
	high	75.11	72.02	0.83
Segment	low	-	86.08	0.93
	mid	-	77.53	0.92
	high	-	69.14	0.91
Keyboard-Typo	low	85.06	76.93	0.92
	mid	62.41	60.21	0.88
	high	40.99	44.16	0.84
Natural Noise	low	85.34	78.43	0.92
	mid	65.36	65.60	0.91
	high	50.06	56.31	0.90
Phonetic	low	90.62	87.40	0.93
	mid	89.09	84.75	0.92
	high	88.95	82.80	0.91
Visual	low	80.52	53.07	0.86
	mid	48.14	35.26	0.64
	high	22.44	34.37	0.48

Table 6: Attacks against unshielded model.

Train Test \	level	FS	IS	INT	DIS	TRUN	SEG	KEY	NAT	PH	VIS	POS	NLI	TC
FS	none											95.57	89.56	0.97
	low	X	✓	✓	✓	✓	✓	✓	✓	✓	✓	84.49	73.05	0.95
	mid											63.48	57.54	0.90
	high											45.72	51.73	0.86
IS	none											95.66	88.94	0.96
	low	✓	X	✓	✓	✓	✓	✓	✓	✓	✓	88.29	75.51	0.94
	mid											75.90	69.07	0.91
	high											73.68	68.54	0.90
INT	none											95.65	88.90	0.96
	low	✓	✓	X	✓	✓	✓	✓	✓	✓	✓	87.54	84.27	0.95
	mid											57.58	74.92	0.93
	high											19.44	61.07	0.84
DIS	none											95.69	89.42	0.96
	low	✓	✓	✓	X	✓	✓	✓	✓	✓	✓	86.00	80.00	0.94
	mid											64.39	70.98	0.91
	high											48.65	66.60	0.89
TRUN	none											95.49	89.17	0.96
	low	✓	✓	✓	✓	X	✓	✓	✓	✓	✓	89.98	84.55	0.84
	mid											81.23	81.97	0.83
	high											79.38	81.62	0.82
SEG	none											-	89.02	0.96
	low	✓	✓	✓	✓	✓	X	✓	✓	✓	✓	-	85.38	0.96
	mid											-	76.92	0.95
	high											-	62.83	0.95
KEY	none											95.61	88.80	0.96
	low	✓	✓	✓	✓	✓	✓	X	✓	✓	✓	87.71	80.47	0.95
	mid											68.64	70.69	0.94
	high											46.51	61.83	0.92
NAT	none											95.72	88.67	0.96
	low	✓	✓	✓	✓	✓	✓	✓	X	✓	✓	88.17	81.27	0.96
	mid											72.30	73.05	0.96
	high											56.78	67.40	0.96
PH	none											95.30	88.95	0.96
	low	✓	✓	✓	✓	✓	✓	✓	✓	X	✓	89.74	87.54	0.96
	mid											87.82	86.27	0.95
	high											87.72	85.34	0.95
VIS	none											95.72	89.02	0.96
	low	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	85.18	70.77	0.93
	mid											58.94	48.80	0.85
	high											24.99	40.22	0.75

Table 7: Adversarial training: leave-one-out.

Test \ Train	level	FS	IS	INT	DIS	TRUN	KEY	NAT	PH	VIS
Clean	-	95.17	95.18	95.04	95.44	95.48	95.40	95.40	96.06	95.66
FS	low	88.49	81.68	80.01	82.63	81.85	81.45	80.93	82.16	83.10
	mid	75.73	58.04	54.04	62.17	59.95	57.69	59.62	56.91	60.38
	high	62.27	40.95	36.11	48.06	44.84	40.58	44.40	38.79	42.44
IS	low	87.97	91.98	86.06	86.68	85.65	87.09	85.63	86.10	87.70
	mid	74.23	87.24	71.42	74.77	72.93	74.06	70.94	70.61	74.61
	high	71.72	86.24	69.20	73.20	70.47	71.84	68.54	67.67	72.73
INT	low	79.23	80.38	92.72	81.82	82.05	86.88	82.27	79.26	86.72
	mid	39.12	42.02	88.41	57.68	54.17	59.29	53.62	44.37	60.84
	high	10.46	11.93	80.96	27.95	24.74	29.55	47.71	15.88	22.45
DIS	low	82.29	82.45	84.68	93.31	85.91	85.43	85.72	84.27	85.59
	mid	56.75	60.08	62.94	89.23	67.55	66.68	67.18	60.33	64.77
	high	40.30	45.11	47.45	86.13	53.89	53.37	54.08	42.87	49.31
TRUN	low	88.17	87.50	88.34	88.97	94.58	89.45	90.03	88.63	88.80
	mid	77.18	76.02	78.33	79.58	93.27	80.58	82.33	76.79	78.98
	high	74.91	73.84	76.28	77.90	92.99	78.64	80.84	74.33	76.83
KEY	low	83.70	83.16	85.17	85.36	85.30	90.11	85.97	84.11	86.99
	mid	60.52	60.99	63.81	65.23	64.07	73.45	67.06	60.62	66.53
	high	36.92	38.50	44.95	44.25	45.64	62.99	47.71	37.72	46.93
NAT	low	83.92	82.57	85.21	85.13	84.74	85.83	88.52	84.41	86.48
	mid	65.05	62.50	65.83	68.13	66.39	67.97	80.98	64.59	67.51
	high	47.83	47.01	52.39	53.52	54.10	54.26	66.76	48.79	53.25
PH	low	88.82	88.66	88.89	89.40	89.21	88.86	89.81	94.89	89.47
	mid	86.68	86.91	87.12	87.64	87.36	86.93	88.07	94.39	87.67
	high	86.74	86.79	87.00	87.51	87.26	86.85	88.00	94.36	87.56
VIS	low	82.07	82.13	83.10	82.64	82.61	84.21	82.26	81.20	87.95
	mid	53.15	54.79	56.05	57.27	55.55	58.06	53.23	49.32	66.58
	high	23.90	26.06	22.19	30.31	29.04	30.43	26.25	23.66	31.38

Table 8: Part-of-Speech tagging adversarial training: 1-1.

Test \ Train	level	FS	IS	INT	DIS	TRUN	SEG	KEY	NAT	PH	VIS
Clean	-	87.54	-	88.29	88.59	88.91	89.90	89.17	89.12	90.24	-
FS	low	83.04	-	64.65	63.28	60.38	62.67	69.46	68.46	66.87	-
	mid	78.58	-	47.62	48.15	42.21	46.05	52.63	50.81	48.11	-
	high	76.96	-	42.75	44.52	39.16	41.55	47.77	46.21	41.53	-
IS	low	81.31	-	71.32	66.40	59.81	63.26	72.23	72.25	66.40	-
	mid	78.82	-	64.82	58.42	51.17	53.63	63.21	64.24	57.16	-
	high	78.08	-	64.06	58.25	50.86	53.53	62.72	62.89	56.61	-
INT	low	76.22	-	85.83	72.49	72.97	72.78	83.73	80.83	74.83	-
	mid	58.99	-	82.61	53.87	51.09	48.45	69.53	62.84	51.53	-
	high	43.22	-	80.76	39.93	36.18	36.60	48.77	40.91	37.07	-
DIS	low	79.14	-	76.27	86.56	67.51	72.52	78.96	77.77	72.65	-
	mid	72.47	-	67.43	84.70	57.60	56.88	69.72	64.85	57.03	-
	high	69.45	-	63.90	84.16	54.50	48.25	65.29	58.44	48.92	-
TRUN	low	81.63	-	84.31	79.66	88.15	80.02	86.35	84.79	80.46	-
	mid	77.87	-	82.45	75.86	87.52	76.11	84.08	81.83	76.19	-
	high	77.25	-	82.46	75.10	87.44	75.79	83.80	81.76	75.80	-
SEG	low	82.32	-	84.32	83.75	84.00	89.07	86.15	85.53	85.69	-
	mid	68.85	-	76.41	75.84	77.33	87.54	80.28	79.37	78.14	-
	high	50.94	-	64.98	68.11	71.36	86.42	73.39	73.19	71.88	-
KEY	low	74.23	-	76.90	71.38	69.95	73.10	86.63	81.04	74.46	-
	mid	57.37	-	62.86	55.62	54.30	58.67	82.98	70.74	56.98	-
	high	45.87	-	52.26	46.29	44.75	47.07	79.82	61.76	44.54	-
NAT	low	77.87	-	78.32	73.62	73.50	75.51	82.39	87.67	76.47	-
	mid	67.98	-	67.98	62.27	60.10	62.97	74.25	85.45	62.85	-
	high	59.73	-	60.81	55.16	53.18	55.41	69.52	84.06	54.89	-
PH	low	85.68	-	85.98	84.23	85.36	86.53	87.50	87.80	89.93	-
	mid	84.25	-	84.21	80.98	82.67	84.17	85.98	86.50	89.40	-
	high	83.07	-	82.71	80.28	81.68	82.68	84.74	85.42	89.19	-
VIS	low	59.79	-	72.33	55.74	56.09	56.91	70.65	66.32	55.34	-
	mid	41.82	-	50.95	37.87	37.01	36.38	45.21	41.84	36.31	-
	high	37.42	-	39.08	33.81	34.26	34.51	36.04	35.37	34.10	-

Table 9: Natural language inference adversarial training: 1-1.

Test \ Train	level	FS	IS	INT	DIS	TRUN	SEG	KEY	NAT	PH	VIS
Clean	-	0.96	0.96	0.96	0.97	0.97	0.98	0.97	0.96	0.97	0.96
FS	low	0.94	0.94	0.94	0.94	0.94	0.95	0.95	0.93	0.95	0.93
	mid	0.92	0.90	0.87	0.88	0.87	0.87	0.89	0.87	0.87	0.88
	high	0.90	0.86	0.80	0.81	0.79	0.79	0.83	0.80	0.77	0.83
IS	low	0.94	0.95	0.94	0.94	0.94	0.95	0.94	0.93	0.94	0.93
	mid	0.92	0.94	0.89	0.90	0.89	0.89	0.91	0.88	0.89	0.89
	high	0.91	0.94	0.88	0.90	0.88	0.88	0.90	0.87	0.88	0.89
INT	low	0.95	0.95	0.96	0.96	0.96	0.96	0.96	0.95	0.96	0.95
	mid	0.92	0.92	0.95	0.92	0.92	0.91	0.94	0.90	0.91	0.91
	high	0.81	0.80	0.91	0.80	0.76	0.75	0.81	0.76	0.72	0.83
DIS	low	0.94	0.95	0.95	0.96	0.94	0.96	0.95	0.94	0.95	0.94
	mid	0.91	0.91	0.91	0.96	0.90	0.90	0.92	0.90	0.89	0.90
	high	0.88	0.88	0.88	0.95	0.86	0.83	0.89	0.86	0.83	0.88
TRUN	low	0.95	0.95	0.96	0.96	0.97	0.96	0.97	0.94	0.96	0.95
	mid	0.94	0.94	0.95	0.94	0.97	0.95	0.96	0.93	0.95	0.94
	high	0.94	0.94	0.95	0.94	0.97	0.95	0.96	0.93	0.95	0.94
SEG	low	0.96	0.96	0.96	0.96	0.97	0.97	0.97	0.96	0.97	0.95
	mid	0.95	0.95	0.95	0.96	0.96	0.97	0.96	0.95	0.96	0.94
	high	0.94	0.94	0.94	0.95	0.95	0.97	0.95	0.93	0.95	0.93
KEY	low	0.95	0.95	0.95	0.95	0.96	0.96	0.96	0.95	0.96	0.95
	mid	0.92	0.92	0.93	0.93	0.93	0.94	0.95	0.92	0.93	0.92
	high	0.88	0.88	0.91	0.89	0.90	0.89	0.95	0.89	0.87	0.90
NAT	low	0.96	0.96	0.96	0.96	0.97	0.97	0.97	0.96	0.97	0.95
	mid	0.95	0.95	0.96	0.96	0.96	0.97	0.97	0.96	0.96	0.95
	high	0.95	0.95	0.95	0.95	0.96	0.96	0.96	0.96	0.96	0.94
PH	low	0.96	0.96	0.96	0.96	0.97	0.97	0.97	0.95	0.97	0.95
	mid	0.95	0.95	0.95	0.96	0.96	0.97	0.96	0.95	0.97	0.95
	high	0.95	0.95	0.95	0.95	0.96	0.96	0.96	0.94	0.97	0.94
VIS	low	0.92	0.93	0.94	0.93	0.93	0.93	0.94	0.91	0.92	0.93
	mid	0.82	0.81	0.86	0.80	0.78	0.77	0.83	0.76	0.71	0.90
	high	0.70	0.69	0.75	0.65	0.62	0.62	0.66	0.64	0.55	0.85

Table 10: Toxic comment adversarial training: 1-1.

# Point-of-Interest Type Inference from Social Media Text

Danae Sánchez Villegas<sup>α</sup> Daniel Preoțiuc-Pietro<sup>β</sup> Nikolaos Aletras<sup>α</sup>

<sup>α</sup> Computer Science Department, University of Sheffield, UK

<sup>β</sup> Bloomberg

{dsanchezvillegas1, n.aletras}@sheffield.ac.uk

dpreoitiucpie@bloomberg.net

## Abstract

Physical places help shape how we perceive the experiences we have there. We study the relationship between social media text and the type of the place from where it was posted, whether a park, restaurant, or someplace else. To facilitate this, we introduce a novel data set of  $\sim 200,000$  English tweets published from 2,761 different points-of-interest in the U.S., enriched with place type information. We train classifiers to predict the type of the location a tweet was sent from that reach a macro F1 of 43.67 across eight classes and uncover the linguistic markers associated with each type of place. The ability to predict semantic place information from a tweet has applications in recommendation systems, personalization services and cultural geography.<sup>1</sup>

## 1 Introduction

Social networks such as Twitter allow users to share information about different aspects of their lives including feelings and experiences from places that they visit, from local restaurants to sport stadiums and parks. Feelings and emotions triggered by performing an activity or living an experience in a Point-of-Interest (POI) can give a glimpse of the atmosphere in that place (Tanasescu et al., 2013).

In particular, the language used in posts from POIs is an important component that contributes toward the place’s identity and has been extensively studied in the context of social and cultural geography (Tuan, 1991; Scollon and Scollon, 2003; Benwell and Stokoe, 2006). Social media posts from a particular location are usually focused on the person posting the content, rather than on providing explicit information about the place. Table 1 displays example Twitter posts from different POIs. Users express their feelings related to a certain

place (‘this places gives me war flashbacks’), comments and thoughts associated with the place they are in (‘few of us dressed appropriately’) or activities they are performing (‘leaving the news station’, ‘on the way to the APCE Annual’).

In this paper, we aim to study the language that people on Twitter use to share information about a specific place they are visiting. Thus, we define the prediction of a POI type given a post (i.e. tweet) as a multi-class classification task using only information available at posting time. Given the text from a user’s post, our goal is to predict the correct type of the location it was posted, e.g. park, bar or shop. Inferring the type of place from a user’s post using linguistic information, is useful for cultural geographers to study a place’s identity (Tuan, 1991) and has downstream geosocial applications such as POI visualisation (McKenzie et al., 2015) and recommendation (Alazzawi et al., 2012; Yuan et al., 2013; Preoțiuc-Pietro and Cohn, 2013; Gao et al., 2015).

Predicting the type of a POI is inherently different to predicting the POI type from comments or reviews. The role of the latter is to provide opinions or descriptions of the places, rather than the activities and feelings of the user posting the text (McKenzie et al., 2015), as illustrated in Table 1. This is also different, albeit related, to the popular task of geolocation prediction (Cheng et al., 2010; Eisenstein et al., 2010; Han et al., 2012; Roller et al., 2012; Rahimi et al., 2015; Dredze et al., 2016), as this aims to infer the exact geographical location of a post using language variation and geographical cues rather than inferring the place’s type. Our task aims to uncover the geographic agnostic features associated with POIs of different types.

Our contributions are as follows: (1) We provide the first study of POI type prediction in computational linguistics; (2) A large data set made out of

<sup>1</sup>Data is available here: <https://archive.org/details/poi-data>

Category	Sample Tweet	Train	Dev	Test	Tokens
Arts & Entertainment	i'm back in central park . this place gives me war flashbacks now lol	40,417	4,755	5,284	14.41
College & University	currently visiting my dream school 😊 ❤️	21,275	2,418	2,884	15.52
Food	Some Breakfast, it's only right! #LA	6,676	869	724	14.34
Great Outdoors	Sorry Southport, Billy is dishing out donuts at #donutfest today. See you next weekend!	27,763	4,173	3,653	13.49
Nightlife Spot	Chicago really needs to step up their Aloha shirt game. Only a few of us dressed "appropriately" tonight. :) 🎉🌴🌺	5,545	876	656	15.46
Professional & Other Places	Leaving the news station after a long day	30,640	3,381	3,762	16.46
Shop & Service	Came to get an old fashioned tape measures and a button for my coat	8,285	886	812	15.31
Travel & Transport	Shoutout to anyone currently on the way to the APCE Annual Event in Louisville, KY! #APCE2018	16,428	2,201	1,872	14.88

Table 1: Place categories with sample tweets and data set statistics.

tweets linked to particular POI categories; (3) Linguistic and temporal analyses related to the place the text was posted from; (4) Predictive models using text and temporal information reaching up to 43.67 F1 across eight different POI types.

## 2 Point-of-Interest Type Data

We define the POI type prediction as a multi-class classification task performed at the social media post level. Given a post  $T$ , defined as a sequence of tokens  $T = \{t_1, \dots, t_n\}$ , the goal is to label  $T$  as one of the  $M$  POI categories. We create a novel data set for POI type prediction containing text and the location type it was posted from as, to the best of our knowledge, no such data set is available. We use Twitter as our data source because it contains a large variety of linguistic information such as expression of thoughts, opinions and emotions (Java et al., 2007; Kouloumpis et al., 2011).

### 2.1 Types of POIs

Foursquare is a location data platform that manages ‘Places by Foursquare’, a database of more than 105 million POIs worldwide. The place information includes verified metadata such as name, geo-coordinates and categories as well as other user-sourced metadata such as tags, comments or photos. POIs are organized into 9 top level primary categories with multiple subcategories. We only focus on 8 primary top-level POI categories since the category ‘Residence’ has a considerably smaller number of tweets compared to the other categories (0.78% tweets from the total). We leave finer-grained place category inference as well as using other metadata for future work since the scope of this work is to study the language of posts associated with semantic type places.

### 2.2 Associating Tweets with POI Types

Twitter users can tag their tweets to the locations they are posted from by linking to Foursquare places.<sup>2</sup> In this way, we collect tweets assigned to the POIs and associated metadata (see Table 1). We select a broad range of locations for our experiments. There is no public list of all Foursquare locations that can be used through Twitter and can be programmatically accessed. Hence, in order to discover Foursquare places that are actually used in tweets, we start with all places found in a 1% sample of the Twitter feed between 31 July 2016 and 24 January 2017 leading us to a total of 9,125 different places. Then, we collect all tweets from these places between 17 August 2016 and 1 March 2018 using the Twitter Search API<sup>3</sup>. We collect the place metadata from the public Foursquare Venues API. This resulted in a total data set of 1,648,963 tweets tagged to a Foursquare place. In order to extract metadata about each location, we crawled the Twitter website to identify the corresponding Foursquare Place ID of each Twitter place. Then, we used the public Foursquare Venues API<sup>4</sup> to download all the place metadata.

### 2.3 Data Filtering

To limit variation in our data, we filter out all non-English tweets and non-US places, as these were very limited in number. We keep POIs with at least 20 tweets and randomly subsample 100 tweets from POIs with more tweets to avoid skewing our data. Our final data set consists of 196,235 tweets from

<sup>2</sup><https://developer.foursquare.com/places>

<sup>3</sup><https://developer.twitter.com/en/docs/tweets/search/guides/tweets-by-place>

<sup>4</sup><https://developer.foursquare.com/overview/venues.html>

2,761 POIs.

## 2.4 Data Split

We create our data split at a location-level to ensure that our models are robust and generalize to locations held-out in training. We split the locations in train (80%), development (10%) and test (10%) sets and assign tweets to one of the three splits based on the location they were posted from (see Table 1 for detailed statistics).

## 2.5 Text Processing

We lower-case text and replace all URLs and mentions of users with placeholders. We preserve emoticons and punctuation and replace tokens that appear in less than five tweets with an ‘unknown’ token. We tokenize text using a Twitter-aware tokenizer (Schwartz et al., 2017).

## 3 Analysis

We first analyze our data set to understand the relationship between location type, language and posting time.

### 3.1 Linguistic Analysis

We analyze the linguistic features specific to each category by ranking unigrams that appear in at least 5 different locations, such that these are representative of the larger POI category rather than a few specific places. Features are normalized to sum up to unit for each tweet, then we compute the (Pearson)  $\chi^2$  coefficient independently between its distribution across posts and the binary category label of the post similar to the approach followed by Maronikolakis et al. (2020) and Preotiuc-Pietro et al. (2019). Table 2 presents the top unigram features for each category.

We note that most top unigrams specific of a category naturally refer to types of places (e.g. ‘campus’, ‘beach’, ‘mall’, ‘airport’) that are part of that category. All categories also contain words that refer to activities that the poster of the tweet is performing or observing while at a location (e.g. ‘camp’ and ‘football’ for College, ‘concert’ and ‘show’ for Arts & Entertainment, ‘party’ for Nightlife Spot, ‘landed’ for Travel & Transport, ‘hike’ for Greater Outdoors). Nightlife Spot and Food categories are represented by types of food or drinks that are typically consumed at these locations. Beyond these typical associations, we highlight that usernames are more likely mentioned in

the Arts & Entertainment category, usually indicating activities involving groups of users, emojis indicative of the user state (e.g. happy emoji in Food places) and adjectives indicative of the user’s surroundings (e.g. ‘beautiful’ in Greater Outdoors places). Finally, we also uncover words indicative of the time the user is at a place, such as ‘tonight’ for Arts & Entertainment, ‘sunset’ for the Greater Outdoors and ‘night’ for Nightlife Spots and Arts & Entertainment.

## 3.2 Temporal Analysis

We further examine the relationship between the time a tweet was posted and the POI type it was posted from. Figure 1 shows the percentage of tweets by day of week (top) and hour of day (bottom).

We observe that tweets posted from the ‘Professional & Other Places’, ‘Travel & Transport’ and ‘College & University’ categories are more prevalent on weekdays, peaking on Wednesday, while on weekends more tweets are posted from the ‘Great Outdoors’, ‘Arts & Entertainment’, ‘Nightlife & Spot’ and ‘Food’ categories when people focus less on professional activities and dedicate more time to leisure as expected. The hour of day pattern follows the daily human activity rhythm, but the differences between categories are less prominent, perhaps with the exception of the ‘Arts & Entertainment’ category peaks around 8PM and ‘Nightlife Spots’ that see a higher percent of tweets in the early hours of the day (between 1-5am) than other categories.

## 4 Predicting POI Types of Tweets

### 4.1 Methods

**Logistic Regression** We first experiment with logistic regression using a standard bag of n-grams representation of the tweet (**LR-W**), including unigrams to trigrams weighted using TF-IDF. We identified in the analysis section that temporal information about the tweet may be useful for classification. Hence, to add temporal information extracted from a tweet, we create a 31-dimensional vector encoding the hour of the day and the day of the week it was sent from. We experiment with only using the temporal features (**LR-T**) and in combination with the text features (**LR-W+T**). We use L1 regularization (Hoerl and Kennard, 1970) with hyperparameter  $\alpha = .01$  (selected based on dev set from  $\{.001, .01, .1\}$ ).

Arts		College		Food		Outdoors		Nightlife		Professional		Shop		Travel	
Feature	$\chi^2$	Feature	$\chi^2$	Feature	$\chi^2$	Feature	$\chi^2$	Feature	$\chi^2$	Feature	$\chi^2$	Feature	$\chi^2$	Feature	$\chi^2$
concert	167.20	campus	298.74	chicken	375.52	beach	591.81	#craftbeer	425.97	school	87.46	mall	462.03	airport	394.20
museum	152.14	college	266.63	#nola	340.64	noen	239.00	🎉	311.68	students	79.93	store	403.00	✈️	343.30
show	134.39	university	155.65	lunch	255.98	hike	227.91	beer	203.57	grade	66.05	shopping	359.00	flight	292.94
night	104.48	class	112.23	fried	216.49	lake	193.58	bar	93.90	vote	65.80	shop	132.39	hotel	168.38
tonight	80.76	semester	103.19	dinner	203.65	park	165.92	🍷	67.00	our	63.12	🌸	126.07	conference	141.74
game	73.56	football	59.24	🍔	195.41	island	151.45	🎵	56.94	jv	60.64	💳	95.32	landed	118.05
art	69.77	student	57.86	pizza	190.83	sunset	142.44	dj	56.56	church	52.97	apple	88.74	plane	88.42
USER	66.14	classes	57.37	shrimp	188.77	hiking	137.74	tonight	53.39	hs	50.63	market	76.60	bound	78.43
zoo	66.09	students	56.98	👉	179.39	beautiful	109.45	ale	52.62	senior	50.05	auto	73.52	heading	62.09
baseball	62.90	camp	44.19	😊	151.00	bridge	108.56	party	51.14	ss	44.46	stock	72.31	headed	57.12

Table 2: Unigrams associated with each category, sorted by  $\chi^2$  value computed between the normalized frequency of each feature and the category label across all tweets in the training set ( $p < 0.001$ ).

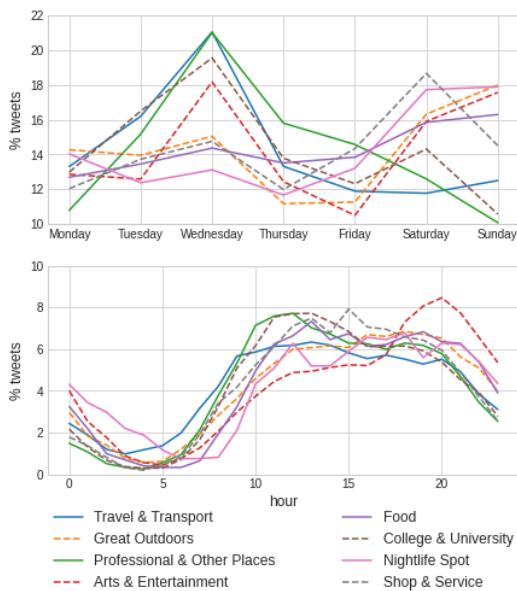


Figure 1: Percentage of tweets by day of week (top) and by hour of day (bottom).

**BiLSTM** We train models based on bidirectional Long-Short Term Memory (LSTM) networks (Hochreiter and Schmidhuber, 1997), which are popular in text classification tasks. Tokens in a tweet are mapped to embeddings and passed through the two LSTM networks, each processing the input in opposite directions. The outputs are concatenated and passed to the output layer using a softmax activation function (**BiLSTM**). We extend the BiLSTM to encode temporal one-hot representation by: (a) concatenating the temporal vector to the tweet representation (**BiLSTM-TC**); and (b) projecting the time vector into a dense representation using a fully connected layer which is added to the tweet representation before passing it through the output layer using a softmax activation function (**BiLSTM-TS**). We use 200-dimensional GloVe embeddings (Pennington et al., 2014) pre-trained on Twitter data. The maximum sequence length is set to 26, covering 95% of the

tweets in the training set. The LSTM size is  $h = 32$  where  $h \in \{32, 64, 100, 300\}$  with dropout  $d = 0.5$  where  $d \in \{.2, .5\}$ . We use Adam (Kingma and Ba, 2014) with default learning rate, minimizing cross-entropy using a batch size of 32 over 10 epochs with early stopping.

**BERT** Bidirectional Encoder Representations from Transformers (BERT) is a pre-trained language model based on transformer networks (Vaswani et al., 2017; Devlin et al., 2019). BERT consists of multiple multi-head attention layers to learn bidirectional embeddings for input tokens. The model is trained on masked language modeling, where a fraction of the input tokens in a given sequence is replaced with a mask token, and the model attempts to predict the masked tokens based on the context provided by the non-masked tokens in the sequence. We fine-tune BERT for predicting the POI type of a tweet by adding a classification layer with softmax activation function on top of the Transformer output for the ‘classification’ [CLS] token (**BERT**). Similarly to the previous models, we extend BERT to make use of the time vector in two ways, by concatenating (**BERT-TC**), and by adding it (**BERT-TS**) to the output of the Transformer before passing it to through the classification layer with softmax activation function. We use the base model (12-layer, 110M parameters) trained on lower-cased English text. We fine-tune it for 2 epochs with a learning rate  $l = 2e^{-5}$ ,  $l \in \{2e^{-5}, 3e^{-5}, 5e^{-5}\}$  and a batch size of 32.

## 4.2 Results

Table 3 presents the results of POI type prediction measured using accuracy, macro F1, precision and recall across three runs. In general, we observe that we can predict POI types of tweets with good accuracy, considering the classification is across eight relatively well balanced classes.

Model	Acc	F1	P	R
<b>Major. Class</b>	26.89	5.30	3.36	12.50
<b>Random</b>	13.63	12.64	13.63	15.68
<b>LR-T</b>	27.93	14.01	15.78	16.06
<b>LR-W</b>	43.04	37.33	37.06	38.03
<b>LR-W+T</b>	43.73	37.83	37.68	38.37
<b>BiLSTM</b>	44.38	35.77	45.29	33.78
<b>BiLSTM-TC</b>	44.01	38.07	41.51	36.46
<b>BiLSTM-TS</b>	44.72	38.26	42.91	36.30
<b>BERT</b>	48.89	<b>43.67</b>	<b>48.44</b>	<b>41.33</b>
<b>BERT-TC</b>	46.13	41.19	46.81	39.03
<b>BERT-TS</b>	<b>49.17</b>	43.47	48.40	41.26

Table 3: Accuracy (Acc), Macro-F1 Score (F1), Precision macro (P), and Recall macro (R) for POI type prediction (all std. dev < 0.01). Best results are in bold.

Best results are obtained using BERT-based models (BERT, BERT-TC and BERT-TS), with the highest accuracy of 49.17 (compared to 26.89 majority class) and highest macro-F1 of 43.67 (compared to 12.64 random). We observe that BERT models outperform both BiLSTM and linear methods across all metrics, with over 4% improvement in accuracy and 5 points F1. The BiLSTM models perform marginally better than the linear models. Temporal features alone are marginally useful when models are evaluated using accuracy (+0.28 BERT, +0.34 for BiLSTMs, +0.69 for LR) and perform similarly on F1, with the notable exception of the BiLSTM models. We find that adding these features is more beneficial than concatenating them, with concatenation hurting performance on accuracy for both BiLSTM and BERT.

Figure 2 shows the confusion matrix of our best performing model, BERT, according to the macro-F1 score. The confusion matrix is normalized over the actual values (rows). The category ‘Arts & Entertainment’ has the greatest percentage (62%) of correctly classified tweets, followed by the ‘Great Outdoors’ category with 54%, and the ‘College & University’ category with 44%. On the other hand, the categories ‘Nightlife Spot’ and ‘Shop & Service’ have the lowest results, where 30% of the tweets predicted as each of these classes is correctly classified. Most common error is when the model classifies tweets from the category ‘College & University’ as ‘Professional & Other Places’, as tweets from these places contain similar terms such as ‘students’ or ‘class’.

## 5 Conclusion

We presented the first study on predicting the POI type a social media message was posted from

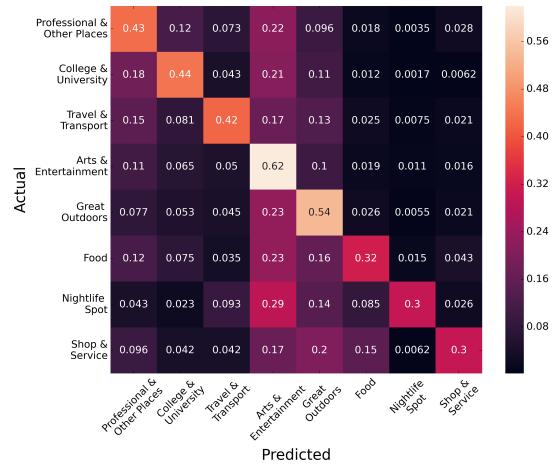


Figure 2: Confusion Matrix of the best performing model (BERT).

and developed a large-scale data set with tweets mapped to their POI category. We conducted an analysis to uncover features specific to place type and trained predictive models to infer the POI category using only tweet text and posting time with accuracy close to 50% across eight categories. Future work will focus on using other modalities such as network (Aletras and Chamberlain, 2018; Tsakalidis et al., 2018) or image information (Vempala and Preoțiuc-Pietro, 2019; Alikhani et al., 2019) and prediction at a more granular level of POI types.

## Acknowledgments

DSV is supported by the Centre for Doctoral Training in Speech and Language Technologies (SLT) and their Applications funded by the UK Research and Innovation grant EP/S023062/1. NA is supported by ESRC grant ES/T012714/1.

## References

- Ahmed N Alazzawi, Alia I Abdelmoty, and Christopher B Jones. 2012. What can I do there? Towards the automatic discovery of place-related services and activities. *International Journal of Geographical Information Science* 26(2):345–364.
- Nikolaos Aletras and Benjamin Paul Chamberlain. 2018. Predicting Twitter user socioeconomic attributes with network and language information. In *Proceedings of the 29th Conference on Hypertext and Social Media*, pages 20–24.
- Malihe Alikhani, Sreyasi Nag Chowdhury, Gerard de Melo, and Matthew Stone. 2019. **CITE: A corpus of image-text discourse relations**. In *Proceedings of the 2019 Conference of the North American*

- Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers).* Association for Computational Linguistics, Minneapolis, Minnesota, pages 570–575.
- Bethan Benwell and Elizabeth Stokoe. 2006. *Discourse and identity*. Edinburgh University Press.
- Zhiyuan Cheng, James Caverlee, and Kyumin Lee. 2010. You Are Where You Tweet: A Content-Based Approach to Geo-Locating Twitter Users. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*. CIKM ’10, pages 759–768.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. *BERT: Pre-training of deep bidirectional transformers for language understanding*. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Minneapolis, Minnesota, pages 4171–4186.
- Mark Dredze, Miles Osborne, and Prabhanjan Kambadur. 2016. *Geolocation for twitter: Timing matters*. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, San Diego, California, pages 1064–1069.
- Jacob Eisenstein, Brendan O’Connor, Noah A. Smith, and Eric P. Xing. 2010. *A latent variable model for geographic lexical variation*. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Cambridge, MA, pages 1277–1287.
- Huiji Gao, Jiliang Tang, Xia Hu, and Huan Liu. 2015. Content-aware point of interest recommendation on location-based social networks. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*. AAAI, pages 1721–1727.
- Bo Han, Paul Cook, and Timothy Baldwin. 2012. *Geolocation prediction in social media data by finding location indicative words*. In *Proceedings of COLING 2012*. The COLING 2012 Organizing Committee, Mumbai, India, pages 1045–1062.
- Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.
- Arthur E Hoerl and Robert W Kennard. 1970. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics* 12(1):55–67.
- Akshay Java, Xiaodan Song, Tim Finin, and Belle Tseng. 2007. Why we Twitter: understanding microblogging usage and communities. In *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 Workshop on Web Mining and Social Network Analysis*, pages 56–65.
- Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Efthymios Kouloumpis, Theresa Wilson, and Joanna D Moore. 2011. Twitter sentiment analysis: The good the bad and the omg! In *Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media*. ICWSM, pages 538–541.
- Antonios Maronikolakis, Danae Sánchez Villegas, Daniel Preoṭiuc-Pietro, and Nikolaos Aletras. 2020. Analyzing political parody in social media. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. pages 4373–4384.
- Grant McKenzie, Krzysztof Janowicz, Song Gao, Jiue-An Yang, and Yingjie Hu. 2015. POI pulse: A multi-granular, semantic signature-based information observatory for the interactive visualization of big geosocial data. *Cartographica: The International Journal for Geographic Information and Geovisualization* 50(2):71–85.
- Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. *Glove: Global vectors for word representation*. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, pages 1532–1543.
- Daniel Preoṭiuc-Pietro and Trevor Cohn. 2013. Mining user behaviours: a study of check-in patterns in location based social networks. In *Proceedings of the 5th annual ACM Web Science Conference*. Web Science, pages 306–315.
- Daniel Preoṭiuc-Pietro, Mihaela Gaman, and Nikolaos Aletras. 2019. Automatically identifying complaints in social media. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. pages 5008–5019.
- Afshin Rahimi, Duy Vu, Trevor Cohn, and Timothy Baldwin. 2015. *Exploiting text and network context for geolocation of social media users*. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Denver, Colorado, pages 1362–1367.
- Stephen Roller, Michael Speriosu, Sarat Rallapalli, Benjamin Wing, and Jason Baldridge. 2012. *Supervised text-based geolocation using language models on an adaptive grid*. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics, Jeju Island, Korea, pages 1500–1510.

H. Andrew Schwartz, Salvatore Giorgi, Maarten Sap, Patrick Crutchley, Lyle Ungar, and Johannes Eichstaedt. 2017. **DLATK: Differential language analysis ToolKit**. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics, Copenhagen, Denmark, pages 55–60.

Ron Scollon and Suzie Wong Scollon. 2003. *Discourses in place: Language in the material world*. Routledge.

Vlad Tanasescu, Christopher B Jones, Gualtiero Colombo, Martin J Chorley, Stuart M Allen, and Roger M Whitaker. 2013. The personality of venues: Places and the five-factors ('big five') model of personality. In *Fourth IEEE International Conference on Computing for Geospatial Research and Application*. pages 76–81.

Adam Tsakalidis, Nikolaos Aletras, Alexandra I Cristea, and Maria Liakata. 2018. Nowcasting the stance of social media users in a sudden vote: The case of the Greek Referendum. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. pages 367–376.

Yi-Fu Tuan. 1991. Language and the making of place: A narrative-descriptive approach. *Annals of the Association of American geographers* 81(4):684–696.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. pages 5998–6008.

Alakananda Vempala and Daniel Preoțiuc-Pietro. 2019. **Categorizing and inferring the relationship between the text and image of twitter posts**. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Florence, Italy, pages 2830–2840.

Quan Yuan, Gao Cong, Zongyang Ma, Aixin Sun, and Nadia Magnenat Thalmann. 2013. Time-aware point-of-interest recommendation. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*. pages 363–372.

# Reconstructing Event Regions for Event Extraction via Graph Attention Networks

Pei Chen<sup>1\*</sup>, Hang Yang<sup>2,3</sup>, Kang Liu<sup>2,3</sup>,  
Ruihong Huang<sup>1</sup>, Yubo Chen<sup>2,3</sup>, Taifeng Wang<sup>4</sup>, and Jun Zhao<sup>2,3</sup>

<sup>1</sup> Texas A&M University, College Station, TX

<sup>2</sup> Institute of Automation, Chinese Academy of Sciences, Beijing, China

<sup>3</sup> University of Chinese Academy of Sciences, Beijing, China

<sup>4</sup> Ant Group, Hangzhou, China

{chenpei, huangrh}@tamu.edu, taifeng.wang@antgroup.com

{hang.yang, kliu, yubo.chen, jzhao}@nlpr.ia.ac.cn

## Abstract

Event information is usually scattered across multiple sentences within a document. The local sentence-level event extractors often yield many noisy event role filler extractions in the absence of a broader view of the document-level context. Filtering spurious extractions and aggregating event information in a document remains a challenging problem. Following the observation that a document has several relevant event regions densely populated with event role fillers, we build graphs with candidate role filler extractions enriched by sentential embeddings as nodes, and use graph attention networks to identify event regions in a document and aggregate event information. We characterize edges between candidate extractions in a graph into rich vector representations to facilitate event region identification. The experimental results on two datasets of two languages show that our approach yields new state-of-the-art performance for the challenging event extraction task.

## 1 Introduction

Event Extraction (EE), a challenging task in Natural Language Processing, aims to extract key types of information (aka *event roles*, e.g., *perpetrators* and *victims* of an *attack* event) that can represent an event in texts and plays a critical role in downstream applications such as Question Answer (Yang et al., 2003) and Summarizing (Filatova and Hatzivassiloglou, 2004). Existing research on EE mostly focused on sentence-level, such as the evaluation in Automatic Content Extraction (ACE) 2005<sup>1</sup>. However, an event is usually described in

\*Most of the work was done when the first author was a research engineer in the Institute of Automation, CAS.

<sup>1</sup><http://projects.ldc.upenn.edu/ace/>

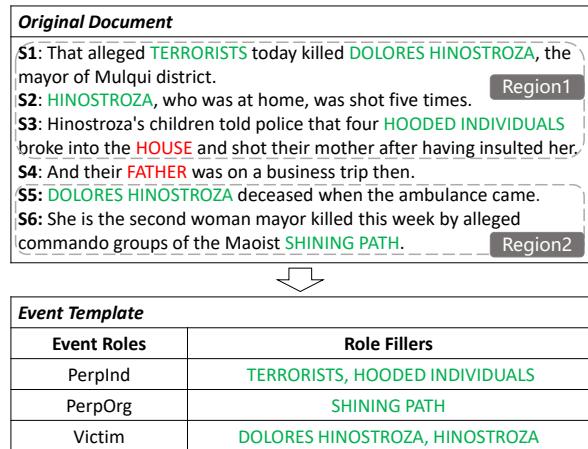


Figure 1: An example of document-level event extraction. We need to extract noun phrases from the document as *role fillers* for the *event roles* in the predefined event template. The uppercased noun phrases in the document are role fillers extracted by the sentence-level extractor. Red phrases are correct while green phrases are noises compared to the standard in the template. There are two *event regions* in the sample document.

multiple sentences in a document. As illustrated in Figure 1, relevant event information (noun phrases in green color) is scattered across the whole document. To extract event information accurately and comprehensively at document-level, it is necessary to understand the wider context spanning over multiple sentences.

The existing approaches for event extraction (EE) often decompose the document-level EE into sentence-level EE, and extract candidate event role fillers from individual sentences one by one. The event role filler extractors often use extraction patterns (Riloff, 1996) or classifiers (Boros et al., 2014) to identify typical local contexts containing a certain type of event role fillers. However, local event role filler extractors often produce many false

candidates, e.g., the red noun phrases shown in the example document of Figure 1.

As shown in the example, one document often mentions a target event multiple times and each time it takes one or more sentences to articulate the event. The target event role fillers tend to be mentioned in several groups of adjacent sentences, and we define those adjacent relevant sentences as different *event regions*. For example, in Figure 1, the document mentions the target event twice in two regions. The correct role fillers are crowding in the first event region  $S_1, S_2, S_3$  and the second one  $S_5, S_6$  respectively. Nevertheless, the sentence-level extractor will extract noise from both the event regions like *HOUSE* from  $S_3$  and irrelevant sentence like *FATHER* in  $S_4$ , destroying the layout of the original regions.

Many previous efforts try to avoid aggregating the noisy candidates by detecting such event regions. The popular approach is to apply sentential classification to filter the sentences and recognize role fillers from the chosen sentences (Patwardhan and Riloff, 2009; Huang and Riloff, 2012). However, these approaches only detect regions at single sentence-level and ignore the crowding of relevant sentences. Also, they also suffer from the accumulative error of sentential classification. For example, they may identify  $S_2$  as a relevant event region but  $S_3$  as irrelevant because they fail to take into account the similarity of  $S_2$  and  $S_3$ . Another solution proposed by Yang *et al.* (2018) tries to detect the primary event description sentence and supplement the missing event roles with fillers from adjacent sentences. This method considers the multiple sentences in an event region but is limited to one region per document. For instance, it may detect  $S_1$  as the primary sentence and supplement it with  $S_2$ , missing the valid items like *SHINING PATH* from region 2. Moreover, it also suffers from the errors selecting primary sentence, and the supplementing strategy is coarse-grained and fails to take into account every candidate filler individually.

We build a graph for each document to directly model the multiple event regions in a document, each region potentially consisting of multiple sentences. In each document graph, the nodes are candidate event role fillers and we insert an edge between two nodes based on either positional proximity (in adjacent sentences or within the same sentence) or the coreference relation between two candidate extractions. The document graphs capture

sentence similarities and sophisticated discourse connections among the candidate event role fillers to reconstruct the original event regions, which can recognize false event role filler extractions from irrelevant sentences. For example, after identifying the differences between  $S_4$  and adjacent sentences  $S_3$  and  $S_5$ , our model will filter the noisy candidate *FATHER* in  $S_4$ .

Furthermore, constructing document graphs formed by candidate event role fillers and applying graph neural networks will enable recognizing false event role filler extractions within an event region. We employ attentional networks on the graphs to reinforce each candidate’s representations by global contextual information and then classify the candidates in a fine-grained manner. Specifically, we characterize the edges into vector representations with rich features to control the information flowing between any two nodes. For instance, this mechanism will be likely to recognize that it is a murder event based on the sentential contexts of sentences  $S_2$  and  $S_3$ , and therefore determine that the candidate extraction *HOUSE* is a false extraction because the Targets of a murder are individuals most commonly, but not physical targets or buildings.

We evaluate our approach on two document-level event extraction datasets: the MUC-4 dataset and a newly created dataset CFEED<sup>2</sup>. Experimental results show that the proposed approach successfully reconstructs 70% of the event regions and yields new state-of-the-art performance for event extraction on both datasets. In summary, the main contributions of this paper are as follows:

- We propose graphs directly modeling the multiple regions with multiple sentences, which successfully help to reconstruct event regions naturally avoid redundant extractions irrelevant sources.
- We propose an edge-enriched graph attention algorithm that can blend both the local clues and global context to enforce semantic representations for each candidate and help to filter noises in the event regions.
- Experimental results show that our method outperforms the existing state-of-the-arts on two datasets with different languages, including a public English MUC-4 dataset and a large-scale Chinese CFEED dataset.

<sup>2</sup><http://www.nlpr.ia.ac.cn/cip/~liukang/dataset/documentevent1.html>

## 2 Related Work

Sentence-level EE has achieved a lot of advancement in recent work (Chen et al., 2015; Nguyen et al., 2016; Chen et al., 2018) and can be classified into template-based approaches (Jungermann and Morik, 2008; Bjorne et al., 2010; Hogenboom et al., 2016) and statistical approaches. Template-based methods require human-crafted templates to match the events. Most of the statistical methods are supervised and either based on feature engineering (Ahn, 2006; Ji and Grishman, 2008; Liao and Grishman, 2010; Reichart and Barzilay, 2012) or Neural network algorithm (Chen et al., 2015; Nguyen et al., 2016; Chen et al., 2018; Liu et al., 2018; Sha et al., 2018; Liu et al., 2018). However, these supervised methods rely on intensive manual annotations. To alleviate this problem, many weak supervised methods (Chen et al., 2017; Zeng et al., 2018) have arisen and achieved good performance in ACE 2005 evaluation.

However, most of the time, people care about the events discussed across a whole document. So research on document-level EE also prevails. Traditionally, pattern-based and classifier-based methods are popular to solve this task. Systems like AutoSlog (Riloff et al., 1993) and AutoSlog-TS (Riloff, 1996) directly applied regular patterns to extract role fillers. Many works (Patwardhan and Riloff, 2007, 2009; Huang and Riloff, 2011, 2012; Boros et al., 2014) relied on feature-based classifiers to distinguish candidate role fillers from texts and achieved better performance. Until recent years, researchers (Hsi, 2018; Yang et al., 2018; Zheng et al., 2019) began to utilize multiple neural-based methods to solve the task. Notably, among the document-level EE research, some works (Patwardhan and Riloff, 2009; Huang and Riloff, 2012; Yang et al., 2018) have noticed the importance of identifying event regions to improve performance.

Traditional neural networks such as Convolutional Neural Networks and Recursive Neural Networks are hard to deal with graphical data structures, so many graph-based neural networks (GNNs) emerge (Gori et al., 2005; Bruna et al., 2013; Kipf and Welling, 2016). In order to deal with graphs with different edge types, relational GNNs (Schlichtkrull et al., 2018; Marcheggiani and Titov, 2017; Vashisht et al., 2019; Bastings et al., 2017) try to use separate weights for different edges. However, one limitation of these GNNs is that the weights are fixed for all

neighbors. So Veličković et al. (2017) leveraged masked attentional layers (GATs) to learn adaptive weights for different neighbors. By now, some works (Schlichtkrull et al., 2018; Vashisht et al., 2019) have successfully applied GNNs to model the document-level information within texts and achieved state-of-the-art performance. Our model is distinguishing because we not only utilize these recent advances but also turns the relational edges to feature-enriched nodes and extends GATs on such heterogeneous graphs.

## 3 Fine-grained Filtering Framework

### 3.1 Overall Framework

Our method for document-level Event Extraction follows three main procedures.

**Extracting role candidates by sentence-level event extractor (SEE):** Given a document, we disintegrate it into a series of sentences and apply sentence-level event extractors to identify candidate role fillers.

**Constructing graphs to model event regions:** Based on the primitive results from the last step and the properties of event regions, we build graphs to capture both the local clues and global context among those candidates.

**Selecting role fillers via edge-enriched graph attention networks (EE-GAT):** We encode the different edges into vectors and then leverage the attention mechanism on the edge-enriched graphs to update the nodes' representations. After that, we feed the candidates to classifiers for filtering.

### 3.2 Extracting Role Candidates by Sentence-level Event Extractor

Sentence-level Event Extractor aims at extracting event roles from each sentence in a document. We reproduce the SEE introduced by Yang et al. (2018) and employ BiLSTM-CRF to identify candidates from each sentence. The model uses the word embedding as the input features, and this method is compatible with both the English and Chinese corpus.

### 3.3 Constructing Graphs to Model Event Regions

For each document, we want to utilize the observed event region information in our model. As discussed before, the original event region information of the candidates from the SEE is destroyed. So we make use of the properties of the original

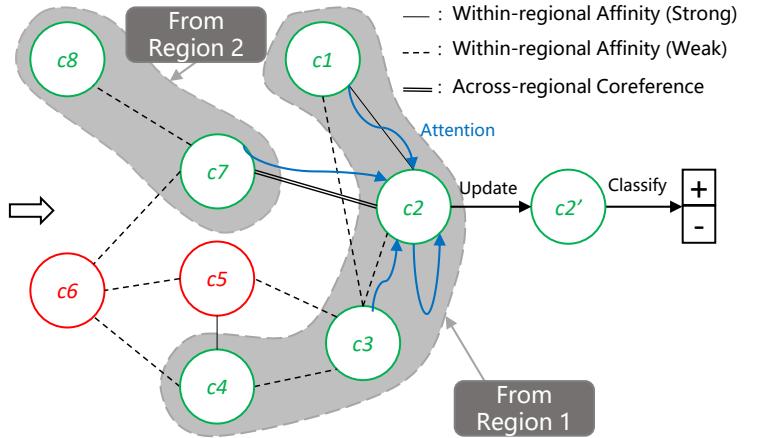
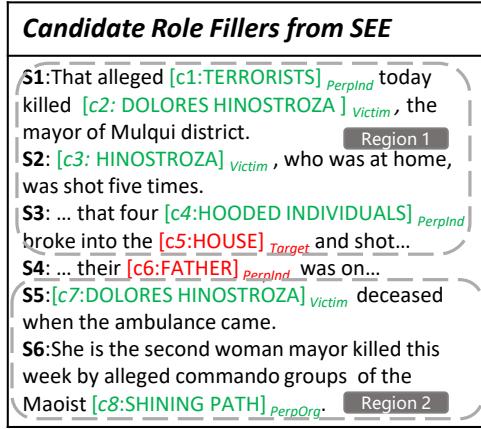


Figure 2: The overall framework of fine-grained filtering framework. 8 candidate role fillers ( $c_1 - c_8$ ) with sentential clues and specific role types are extracted by SEE as nodes. 3 types of edges are defined to connect those nodes: within-regional affinity (Strong), within-regional affinity (weak), across-regional coreference. Then we employ edge-enriched attention mechanism to update the representation of each candidate for classification, like node  $c_2'$  from  $c_2$ . Ideally, the framework will filter noisy candidates  $c_5, c_6$  and reconstruct the original two event regions.

event regions and, according to them, build a graph to link those candidates. Specifically, we first take each candidate role filler as the node in the graph. These nodes can easily take rich candidates' rich features as initial representation, such as the entity embeddings and the local sentential information. For example, in Figure 2, we extract 8 candidate role fillers with specific role type from a document using the aforementioned SEE. We mark them as  $c_1 - c_8$  and regard them as the nodes.

As we know from the property of event regions, the correct role fillers tend to crowd within the same or adjacent sentences, such as  $c_1, c_2, c_3$  and  $c_4$  in Figure 2. Also, one event may be mentioned by multiple event regions, and there can be coreferential role filler across these regions, like  $c_2$  and  $c_7$ . We employ such properties of event regions to construct the graphs so as to utilize regional information. In detail, we define the following 2 types of relations (3 types of edges) in the graphs:

**Within-regional Affinity** When two candidates appear in the same or adjacent sentences, they have a within-regional affinity. We use such affinities to model the phenomenon that multiple event role fillers tend to crowd in an event region. When one candidate filler in the region has high confidence to be a positive one, other candidates can share this confidence and vice versa. Furthermore, we distinguish the same sentence affinity from the adjacent sentences affinity using different edges because we believe such affinity is stronger within the same sentence. For instance, in Figure 2, we assign  $c_1$  and  $c_2$  with strong within-regional affinity since

they are both in  $S_1$ , and use a single solid line to represent this affinity. And we assign  $c_6$  and  $c_7$  with the weak within-regional affinity because they occur in adjacent sentences  $S_4$  and  $S_5$  respectively. A single dotted line is used to illustrate it. The weak affinity may have less confidence sharing and help filter noisy candidate  $c_6$  while keeping  $c_7$ .

**Across-regional Coreference** When two candidates are the same to each other lexically and also recognized as the same event role type, we assume that they have a coreference relationship. When these two coreferential candidates are not in the same or adjacent sentences (they do not have within-regional affinity), we assign them with across-regional coreference so as to bridge different regions. This is because a document usually mentions the target event in multiple event regions, and the same event role fillers may repeat in these regions. We connect these regions by utilizing such cross-region coreference relationships. Such connections will help exchange semantic information and share classification confidence among different regions. Here in Figure 2, we assign  $c_2$  and  $c_7$  with across-regional coreference relationship and use a double solid line to represent corresponding edge in the graph.

Although the constructed graphs do not precisely demonstrate the original event regions, the GNNs models will synthesize comprehensive context from such connections to enforce each candidate's representations, identify the noises, and reconstruct the original regions as a result.

### 3.4 Selecting Role Fillers via Edge-enriched Graph Attention Networks

After building graphs from the documents, we classify the nodes via supervised learning. We first encode the nodes and edges into vectors and then apply the attention mechanism to update the representation of each node from its neighbors, and finally feed the updated representation into classifiers for filtering.

**Encoding** Each graph is represented by its nodes and edges, as  $G = (C, E)$ , where  $C$  represents nodes and  $E$  represents edges. We first initialize all nodes with their feature representations and get  $C = \{c_1, c_2, \dots, c_n\}, c_i \in R^F$ , where  $c_i$  represents the features of node  $i$ ,  $n$  is the number of nodes and  $F$  is the embedding size for each node. Each node is featured by 4 types of embeddings  $c_i = [w_i, p_i, t_i, s_i]$ , where  $w_i$  is the average word embedding of each candidate entity,  $p_i$  is the position embedding of the candidate with respect to the sentence,  $t_i$  is the embedding of role type, and  $s_i$  is the sentence embedding by averaging all words in the sentence.

For edges, the plain graph attention mechanism does not encode them into vectors. Such a mechanism equally treating the edges suffers from losing the information of distinguishing edges. A popular way to deal with this problem is to use different weights for different edges in the attention operation (Relational GAT, R-GAT). However, R-GAT does not have edge representation nor controls the information flow equally for the same type edges. Our edge-enriched attention model characterizes the edges into vector representations, which can especially control the information between each candidate node pair. Initially, we regard each edge as a new type of node featuring its edge type and make a new set of nodes  $E'$ . For example in Figure 3, we use the new node  $e_{1,2} \in E'$  to represents the original within-regional affinity edge between nodes  $c_1$  and  $c_2$ . Here the same type of edges will share the same initial vector representation.

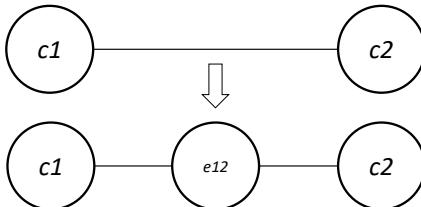


Figure 3: Encoding of Edges

In this way, we construct a new graph  $\tilde{G} = (\tilde{C}, \tilde{E})$  in which all the new edges in the graph are the same, but we have two types of nodes  $\tilde{C} = \{C, E'\}$ , which means the graph is heterogeneous now. To update all nodes in the same attention mechanism, we combine the feature spaces of both the original nodes and new edge-enriched nodes. In this way, any new node within the new graph will have 5 types of embedding:  $\tilde{c}_i = [w_i, p_i, t_i, s_i, e_i]$ , where  $[e_i]$  is the edge type representation. We initialize  $e_i$  as zero vectors for original candidate nodes and the other 4 embeddings as zero vectors for the new edge nodes.

**Updating** Then we update the edge-enriched graph based on GAT proposed by (Veličković et al., 2017). GAT is in essence masked attention operation on graphs. For each layer of graph attention, it updates the representation of node  $\tilde{c}_i$  by computing the linear combinations of its neighbors' normalized attention scores and their corresponding transformed representations:

$$\tilde{c}'_i = \parallel_{h=1}^H \sigma \left( \sum_{j \in \mathcal{N}_i} \alpha_{ij}^h W^h \tilde{c}_j \right) \quad (1)$$

Here we concatenate (signified by  $\parallel$ )  $H$  heads of the attentions results.  $\sigma$  represents the activation functions and  $\mathcal{N}_i$  represents the neighbor nodes of  $\tilde{c}_i$ , including itself. Transformation  $W^h$  is shared for all nodes within each head. We obtain the attention score  $\alpha_{ij}^h$  in head  $h$  as followed:

$$\alpha_{ij}^h = \frac{\exp(\text{LeakyReLU}(a^T (W^h \tilde{c}_i \| W^h \tilde{c}_j)))}{\sum_{k \in N(i)} \exp(\text{LeakyReLU}(a^T (W^h \tilde{c}_i \| W^h \tilde{c}_k)))} \quad (2)$$

Here  $a$  is a single-layer feedforward neural network. We apply two layers of the GAT to update on the graphs. The first layer will exchange the information between candidate nodes and edge nodes, which will characterize the edge representation with the semantic context. Now each edge node will have unique vector representations. Then in the second layer, the candidate nodes will incorporate information from the updated edge nodes, indirectly blend in the features of adjacent candidate nodes in the original graph  $G$ . The enriched edges play the role to control the information flowing between neighbor candidate nodes uniquely.

For comparison, the R-GAT model uses different weights for different edges as followed, where  $\mathcal{R}$  is the set of edge types. Here different edges control

Systems	Event Roles in MUC-4 Dataset					
	PerpInd	PerpOrg	Target	Victim	Weapon	Average
(Riloff, 1996)	33/49/40	53/33/41	54/59/56	49/54/51	38/44/41	45/48/46
(Patwardhan and Riloff, 2009)	51/58/54	34/45/38	43/72/53	55/58/56	57/53/55	48/57/52
(Huang and Riloff, 2011)	48/57/52	46/53/50	51/73/60	56/60/58	53/64/58	51/62/56
(Huang and Riloff, 2012)	54/57/56	55/49/51	55/68/61	<b>63</b> /59/61	62/64/63	58/60/59
(Boros et al., 2014)	53/58/55	56/67/ <b>61</b>	59/63/61	56/55/55	72/65/68	59/61/60
(Yang et al., 2018)	48/60/54	52/74/ <b>61</b>	52/70/59	56/62/59	70/77/74	56/69/61
SEE	35/ <b>77</b> /48	28/ <b>88</b> /42	44/ <b>80</b> /57	38/ <b>83</b> /53	59/ <b>86</b> /70	41/ <b>83</b> /55
GAT	<b>62</b> /52/57	57/53/55	60/61/60	61/58/59	<b>78</b> /78/ <b>78</b>	<b>64</b> /60/62
R-GAT	58/62/ <b>60</b>	57/61/59	60/63/62	57/67/61	71/75/73	61/66/63
EE-GAT	60/59/ <b>60</b>	<b>58</b> /61/60	<b>61</b> /68/ <b>64</b>	62/65/ <b>63</b>	75/75/75	63/66/ <b>65</b>

Table 1: Evaluation on MUC-4 test set, P/R/F1 (Precision/Recall/F1-Score,%).

Event Types	Systems	Event Roles in CFEED Dataset					
		NAME	NUM	BEG	END	ORG	Average
Freeze	(Boros et al., 2014)	71/76/74	56/57/56	<b>77</b> /54/63	83/ <b>80</b> / <b>81</b>	70/80/ <b>75</b>	72/69/70
	(Yang et al., 2018)	<b>83</b> /71/ <b>76</b>	70/49/58	75/67/71	<b>85</b> /65/74	<b>71</b> /67/69	77/64/70
	EE-GAT	68/ <b>82</b> /75	<b>57</b> / <b>63</b> / <b>60</b>	<b>71</b> / <b>77</b> / <b>74</b>	84/79/ <b>81</b>	<b>65</b> / <b>82</b> /72	69/ <b>77</b> / <b>73</b>
Pledge	(Boros et al., 2014)	74/ <b>95</b> /83	60/46/52	<b>68</b> / <b>81</b> /74	74/ <b>30</b> /42	83/ <b>92</b> /87	72/69/70
	(Yang et al., 2018)	<b>84</b> /87/ <b>86</b>	76/54/63	<b>81</b> /72/76	<b>85</b> /28/42	<b>88</b> /82/85	<b>83</b> /64/72
	EE-GAT	77/ <b>95</b> /85	<b>79</b> / <b>55</b> / <b>65</b>	76/78/ <b>77</b>	83/ <b>30</b> / <b>44</b>	84/91/ <b>88</b>	80/ <b>70</b> / <b>75</b>
OW/UW	(Boros et al., 2014)	49/ <b>89</b> /63	63/ <b>65</b> /64	39/ <b>79</b> /52	62/ <b>45</b> /53	—	54/ <b>70</b> /61
	(Yang et al., 2018)	<b>77</b> /70/ <b>73</b>	79/54/64	66/68/67	74/39/51	—	74/58/65
	EE-GAT	66/82/ <b>73</b>	<b>80</b> /60/ <b>68</b>	<b>73</b> / <b>79</b> / <b>76</b>	<b>77</b> /44/ <b>56</b>	—	74/66/ <b>70</b>
Total	(Boros et al., 2014)	65/ <b>87</b> /74	60/56/58	61/71/66	73/ <b>52</b> /61	77/86/ <b>81</b>	66/69/67
	(Yang et al., 2018)	<b>81</b> /76/ <b>78</b>	75/52/61	74/69/71	<b>81</b> /44/57	<b>80</b> /75/77	<b>78</b> /62/69
	EE-GAT	70/86/77	<b>72</b> / <b>59</b> / <b>65</b>	<b>73</b> / <b>78</b> / <b>75</b>	<b>81</b> /51/ <b>63</b>	75/ <b>87</b> / <b>81</b>	74/71/72

Table 2: Evaluation on the CFEED test set, P/R/F1 (Precision/Recall/F1-Score,%).

the information exchange differently. However, this mechanism is not as effective as the enriched edges in our EE-GAT model.

$$c'_i = \left\| \sum_{h=1}^H \sigma \left( \sum_{r \in \mathcal{R}} \sum_{j \in \mathcal{N}_i} \alpha_{ij}^h W^{r,h} c_j \right) \right\| \quad (3)$$

**Classification** After updating the candidate nodes via the two layers multi-head attention mechanism, we need to classify each candidate node as either positive or negative. Now we average the vectors of multiple heads to get the final representation of each node and then project the results into a softmax classification layer.

As a result, we will get the probabilities of the node as either positive or negative. This process is illustrated in equation (4), where  $y_i \in \{0, 1\}$  is the label of node  $i$ ,  $\theta$  represents all the parameters,  $p$  is the probability of  $y_i$  equals to 0 or 1.

$$p(y_i | \tilde{G}; \theta) = \text{softmax} \left( \frac{1}{H} \sum_{h=1}^H \sum_{j \in \mathcal{N}(i)} \alpha_{ij}^h W^h c'_j \right) \quad (4)$$

We train our model to minimize the cross-entropy loss in the data and use the Adam optimization method proposed by Kingma and Ba (2014) to

update the parameters  $\theta$ . The loss function is as followed in equation (4) where  $\hat{y}_i = p(y_i = 1 | G; \theta)$  is the predicted probability of node  $i$  as positive,  $N$  is the number of samples.

$$L(\theta) = - \sum_{i=1}^N (y_i \log \hat{y}_i + (1 - y_i) \log (1 - \hat{y}_i)) \quad (5)$$

## 4 Experiments

### 4.1 MUC-4

**MUC-4** dataset was released by Message Understanding Conferences in 1992. It is about terrorism events and consists of 1700 documents as in Table 4. We follow the same evaluation paradigm as previous work and evaluate the 5 kinds of event roles: *PerpInd*, (individual perpetrator), *PerpOrg* (organizational perpetrator), *Target* (physical target), *Victim* (human target name or description)

Datasets	Event Types	Train	Dev	Test	Total
<b>MUC-4</b>	Terrorism	1300	200	200	1700
	Freeze	589	150	300	1039
	Pledge	3602	300	300	4202
	OW/UW	1303	300	300	1903

Table 3: Statistics of MUC-4 and CFEED

and *Weapon* (instrument id or type). We use head noun matching (e.g. *HINOSTROZA* is considered to match *DOLORES HINOSTROZA*) as before too.

**Baselines** For comparison, we choose the following 6 previous state-of-the-art systems as the baselines for MUC-4.

**Riloff (1996)** automatically produced many domain-specific extraction patterns for role fillers extraction.

**Patwardhan and Riloff (2009)** incorporated both phrasal and sentential evidence to label role fillers. They first used a sentential event recognizer to select sentences and then applied a plausible role-filler recognizer to extract role fillers.

**Huang and Riloff (2011)** designed TIER system to better extract role fillers from Secondary Context, regardless of whether a relevant event is mentioned.

**Huang and Riloff (2012)** defined many features and used SVMs to extract local candidate role fillers and CRF to choose sentences for final results.

**Boros et al. (2014)** utilized domain-relevant word representations as the features of noun phrases and then applied randomized decision trees to identify role fillers. Here we adopt the same idea but use a different classifier MLP. Besides, we use the same node features as in EE-GAT instead of just domain word vectors for comparison with our model.

**Yang et al. (2018)** proposed a document-level EE system following three steps. It first extracted candidate role fillers from each sentence via sequence tagging model; then it applied Convolutional Neural Networks to detect the primary sentence that mentions the target event; finally, it aggregated the candidate role fillers from the primary sentence and supplements the missing even roles from adjacent sentences.

**Experiments on MUC-4** For node representations, we randomly initialize  $p_i, t_i$  as 50-dim vectors and  $e_i$  as 200-dim, and use the 100-dim Glove<sup>3</sup> word embedding for  $w_i, s_i$ . Each layer of the attention mechanism has 8 heads and the learning rate is set as 5e-4. We train on MUC-4 training data for 100 epochs and choose the best model performed on the development set for testing.

We report Precision/Recall/F1-score of the test results for each event role individually and the macro-average over all five roles. The test results

are shown in Table 2. From the table, we have the following observations: (1) In general, our EE-GAT framework achieves the best performance compared with previous state-of-the-art methods. It significantly improves the previous best method by 4.0% (65% vs. 61%) on average F1 score and most of the improvement is contributed by the better precision 7.0% (63% vs. 56%) as opposed to Yang et al. (2018). (2) The SEE results have high recall but very low precision because of the noisy candidates. Plain GAT filters some noises and improves precision a lot. R-GAT and EE-GAT balance the trade-off between precision and recall and achieve a better overall F1 score. (3) In detail, our method achieves the best performance nearly on most of the event roles. We significantly improve the F1 score of 4.0% (60% vs. 56%) in *PerInd* and 3.0% in *Target* (64% vs. 61%) compared to previous best in Huang and Riloff (2012).

## 4.2 CFEED

**CFEED Chinese Financial Event Extraction Dataset** is a larger dataset in Chinese about the major events in the announcements of listed companies. We construct it by the same method proposed by Yang et al. (2018). We crawled the public announcements from sohu.com<sup>4</sup> and the event templates from eastmoney.com<sup>5</sup>, and then align them. We assume that if the key role fillers in a template appear in an announcement, the announcement is describing the event in the template. As in Table 3, it consists of a total of 7144 documents and 3 types of financial events: freezing shares (*freeze*), pledging shares (*pledge*) and overweighting and underweighting shares (*OW&UW*). We defined 5 types of event role in these financial events: shareholder’s name (*NAME*), organization (*ORG*), number of shares (*NUM*), event starting date (*BEG*), event ending date (*END*). Note that the *ORG* is not included in *OW&UW* event.

**Baselines** For comparison, we select the two methods mentioned above as the baselines for CFEED: **Boros et al. (2014)** and **Yang et al. (2018)**.

**Experiments on CFEED** We use the same settings as in MUC-4 to evaluate on the CFEED except that we use the character-level 100-dim embeddings trained on Chinese wiki corpus<sup>6</sup>. We sep-

<sup>4</sup><http://q.stock.sohu.com/index.shtml>

<sup>5</sup><http://choice.eastmoney.com/>

<sup>6</sup><https://github.com/Embedding/Chinese-Word-Vectors>

Statistics	MUC-4			CFEED		
	Gold	SEE	EE-GAT	Gold	SEE	EE-GAT
Avg #Fillers /Doc	8.21	11.17	6.30	11.72	29.95	10.43
Avg #Regions /Doc	1.76	2.86	1.57	2.53	2.21	2.58
Avg #Fillers /Region	5.32	5.54	4.57	5.88	16.94	5.51
Eval for Regions	—	21/87/34	65/70/68	—	16/96/27	68/77/72

Table 4: Distributions of role fillers in the golden data and results of SEE and EE-GAT on the test set of MUC-4 and CFEED. The last row is the evaluation (Precision/Recall/F1-Score,%) of the regions sentence by sentence. The statistics demonstrate the salient Event Regions in golden data and its reconstruction by EE-GAT.

Settings	MUC-4	CFEED			
		Freeze	Pledge	OW&UW	Total
(Yang et al., 2018)	56/69/61	77/64/70	83/64/72	74/58/65	78/62/69
EE-GAT w/ 1st Rel	63/59/61	71/72/71	77/68/72	64/68/66	71/69/70
EE-GAT w/ 1st & 2nd Rel	62/64/63	66/77/71	76/71/73	64/70/67	69/73/71
EE-GAT	63/66/65	69/77/73	80/70/75	74/66/70	74/71/72

Table 5: Effectiveness of the Regional Relations in EE-GAT (Average P/R/F1, Precision/Recall/F1-Score,%). *1st Rel* means strong within-regional affinity and *2nd Rel* means weak within-regional affinity.

arately evaluate the 3 types of events and the results are in Table 3. We can observe that our EE-GAT can achieve the best performance on all the 3 types of events when compared with the baselines. The results verify the robustness of our method in Chinese corpus. Besides, compared with the method in Yang et al. (2018), the major improvement comes from recall rather than precision as on MUC-4. This is because the financial announcement documents in CFEED usually have one main sentence describing the target event, so Yang’s method can achieve high precision by detecting the primary event mention. However, MUC-4 dataset does not have such characteristics.

### 4.3 Reconstructing Event Regions

As in Table 4 about event regions, test if a sentence in the new regions appears in the golden regions and get the evaluation *Precision*, *Recall*, and *F1* scores. We can observe that in both of the datasets: (1) EE-GAT successfully reconstructs 70% of the event regions during the evaluation, which improves about 40% from the SEE results. The detection of the event regions contributes to most of the filtering process. (2) SEE extracted too many noisy role fillers compared to the golden standard. EE-GAT filters many noises and the counts of remaining fillers are similar to the golden standard. (3) The distribution of role fillers and event regions are more close to the golden standard after EE-GAT filtering. In detail, on the gold test sets, there are about 1.76 regions in a document and 5.32 fillers in each region on MUC-4, and 2.53 regions and 5.88 fillers per region on CFEED. However, the event

region distribution diverges after SEE because of the noisy candidates, and we have about 2.86 regions in a document and 5.54 fillers in each region on MUC-4, and 2.21 regions and 16.94 fillers per region on CFEED. Then these statistics recover back to normal after the filtering of EE-GAT, and there are about 1.57 regions in a document and 4.57 fillers in each region on MUC-4, and 2.58 regions and 5.51 fillers per region on CFEED.

### 4.4 Effectiveness of Regional Relations

We set the following control experiments to demonstrate the effectiveness of the regional relations in filtering the noise. We add the three types of edges one by one and test the performance of EE-GAT. As in Table 5, we can observe that the overall performance on all the datasets improves when more types of relations are used. (1) Particularly, even the utilization of strong within-regional affinity (1st Rel) only in EE-GAT achieves slightly better performance compared to the previous state-of-the-art (Yang et al., 2018). (2) Adding the weak within-regional affinity (2nd Rel) further improves the overall performance, especially the average 4.5pp improvement in recall score. (3) And the complete EE-GAT model connecting the multiple event regions achieves even better overall performance. These results demonstrate that the event region relations can capture the global contextual information and help to filter the noisy candidates.

## 5 Conclusion

We propose a fine-grained filtering framework to address the aggregating problem in document-level

event extraction by reconstructing event regions. Our method can filter those noise both in irrelevant sentences and in the event regions and achieve state-of-the-art performance on both the MUC-4 and CFEED datasets. Future work may consider using an end2end model to avoid error propagation from SEE.

## Acknowledgments

This work is supported by the Natural Key RD Program of China (No.2018YFB1005100), the National Natural Science Foundation of China (No.61922085, No.U1936207, No.61806201) and the Key Research Program of the Chinese Academy of Sciences (Grant NO. ZDBS-SSW-JSC006). This work is also supported by CCF-Tencent Open Research Fund, Beijing Academy of Artificial Intelligence (BAAI2019QN0301) and independent research project of National Laboratory of Pattern Recognition.

## References

- David Ahn. 2006. The stages of event extraction. In *The Workshop on Annotating and Reasoning about Time and Events*, pages 1–8.
- Joost Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Sima'an. 2017. Graph convolutional encoders for syntax-aware neural machine translation. *arXiv preprint arXiv:1704.04675*.
- Jari Bjorne, Filip Ginter, Sampo Pyysalo, Jun’ichi Tsujii, and Tapani Salakoski. 2010. Complex event extraction at pubmed scale. *Bioinformatics*, 26(12):i382–i390.
- Emanuela Boros, Romaric Besançon, Olivier Ferret, and Brigitte Grau. 2014. Event role extraction using domain-relevant word representations. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1852–1857.
- Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann Lecun. 2013. Spectral networks and locally connected networks on graphs.
- Yubo Chen, Shulin Liu, Xiang Zhang, Kang Liu, and Jun Zhao. 2017. [Automatically labeled data generation for large scale event extraction](#). In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 409–419, Vancouver, Canada. Association for Computational Linguistics.
- Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event extraction via dynamic multi-pooling convolutional neural networks. In *Proceedings of the ACL*.
- Yubo Chen, Hang Yang, Kang Liu, Jun Zhao, and Yantao Jia. 2018. Collective event detection via a hierarchical and bias tagging networks with gated multi-level attention mechanisms. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1267–1276.
- Elena Filatova and Vasileios Hatzivassiloglou. 2004. Event-based extractive summarization. *Text Summarization Branches Out*.
- Marco Gori, Gabriele Monfardini, and Franco Scarselli. 2005. A new model for learning in graph domains. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, volume 2, pages 729–734. IEEE.
- Frederik Hogenboom, Flavius Frasincar, Uzay Kaymak, Franciska De Jong, and Emiel Caron. 2016. A survey of event extraction methods from text for decision support systems. *Decision Support Systems*, 85(C):12–22.
- Andrew Hsi. 2018. *Event Extraction for Document-Level Structured Summarization*. Ph.D. thesis, Carnegie Mellon University.
- Ruihong Huang and Ellen Riloff. 2011. Peeling back the layers: detecting event role fillers in secondary contexts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1137–1147. Association for Computational Linguistics.
- Ruihong Huang and Ellen Riloff. 2012. Modeling textual cohesion for event extraction. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*.
- Heng Ji and Ralph Grishman. 2008. Refining event extraction through cross-document inference. In *Proceedings of the ACL*, pages 254–262.
- Felix Jungermann and Katharina Morik. 2008. *Enhanced Services for Targeted Information Retrieval by Event Extraction and Data Mining*. Springer Berlin Heidelberg.
- Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks.
- Shasha Liao and Ralph Grishman. 2010. Using document level cross-event inference to improve event extraction. In *Proceedings of the ACL*, pages 789–797.
- Jian Liu, Yubo Chen, Kang Liu, and Jun Zhao. 2018. Event detection via gated multilingual attention mechanism. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

- Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. *arXiv preprint arXiv:1703.04826*.
- Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. Joint event extraction via recurrent neural networks. In *Proceedings of the ACL*, pages 300–309.
- Siddharth Patwardhan and Ellen Riloff. 2007. Effective information extraction with semantic affinity patterns and relevant regions. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 717–727.
- Siddharth Patwardhan and Ellen Riloff. 2009. A unified model of phrasal and sentential evidence for information extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 151–160. Association for Computational Linguistics.
- Roi Reichart and Regina Barzilay. 2012. Multi event extraction guided by global constraints. In *Proceedings of the ACL*, pages 70–79.
- Ellen Riloff. 1996. Automatically generating extraction patterns from untagged text. In *Proceedings of the national conference on artificial intelligence*, pages 1044–1049.
- Ellen Riloff et al. 1993. Automatically constructing a dictionary for information extraction tasks. In *AAAI*, volume 1, pages 2–1. Citeseer.
- Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*, pages 593–607. Springer.
- Lei Sha, Feng Qian, Baobao Chang, and Zhifang Sui. 2018. Jointly extracting event triggers and arguments by dependency-bridge rnn and tensor-based argument interaction. In *Thirty-Second AAAI Conference on Artificial Intelligence*.
- Shikhar Vashisht, Shib Sankar Dasgupta, Swayambhu Nath Ray, and Partha Talukdar. 2019. Dating documents using graph convolution networks.
- Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2017. Graph attention networks.
- Hang Yang, Yubo Chen, Kang Liu, Yang Xiao, and Jun Zhao. 2018. Dcfee: A document-level chinese financial event extraction system based on automatically labeled training data. In *Proceedings of ACL 2018, System Demonstrations*, pages 50–55.
- Hui Yang, Tat-Seng Chua, Shuguang Wang, and Chun-Keat Koh. 2003. Structured use of external knowledge for event-based open domain question answering. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 33–40. ACM.
- Ying Zeng, Yansong Feng, Rong Ma, Zheng Wang, Rui Yan, Chongde Shi, and Dongyan Zhao. 2018. Scale up event extraction learning via automatic training data generation. In *Thirty-Second AAAI Conference on Artificial Intelligence*.
- Shun Zheng, Wei Cao, Wei Xu, and Jiang Bian. 2019. Doc2EDAG: An end-to-end document-level framework for Chinese financial event extraction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 337–346, Hong Kong, China. Association for Computational Linguistics.

# Recipe Instruction Semantics Corpus (RISeC): Resolving Semantic Structure and Zero Anaphora in Recipes

**Yiwei Jiang, Klim Zaporojets, Johannes Deleu, Thomas Demeester, Chris Develder**

Ghent University – imec, IDLab

Ghent, Belgium

{first\\_name.last\\_name}@ugent.be

## Abstract

We propose a newly annotated dataset for information extraction on recipes. Unlike previous approaches to machine comprehension of procedural texts, we avoid a priori pre-defining domain-specific predicates to recognize (e.g., the primitive instructions in MILK) and focus on basic understanding of the expressed semantics rather than directly reduce them to a simplified state representation (e.g., ProPara). We thus frame the semantic comprehension of procedural text such as recipes, as fairly generic NLP subtasks, covering (i) entity recognition (ingredients, tools and actions), (ii) relation extraction (what ingredients and tools are involved in the actions), and (iii) zero anaphora resolution (link actions to implicit arguments, e.g., results from previous recipe steps). Further, our Recipe Instruction Semantic Corpus (RISeC) dataset includes textual descriptions for the zero anaphora, to facilitate language generation thereof. Besides the dataset itself, we contribute a pipeline neural architecture that addresses entity and relation extraction as well as identification of zero anaphora. These basic building blocks can facilitate more advanced downstream applications (e.g., question answering, conversational agents).

## 1 Introduction

Recently, several efforts have aimed at understanding recipe instructions (see Section 2). We consider such recipes as prototypical for procedural texts, for which processing is complex due to the need to (i) understand the ordering of steps (not unlike, e.g., event ordering in news), (ii) solve frequent ellipsis (i.e., zero anaphora) and coreference resolution, and (iii) track the state changes they involve (e.g., ingredients processed/combined to new entities). Especially the latter distinguishes procedural text processing

from more traditional information extraction (e.g., from news).

Most existing works on recipes focus on recognizing pre-defined predicates, typically in the form of a limited set of instruction types (e.g., to convert the recipe to robot instructions) with predefined argument slots to fill. Further, they often rely on an available starting list of ingredients (which may not be available in other procedural text). Hence, current approaches towards recipe understanding make assumptions that are rather domain specific. In contrast, we aim for a more basic and generic structured representation of the procedural text, limiting domain-specific knowledge and building on more general semantic concepts. In particular, we build on semantic concepts as defined in PropBank ([Kingsbury and Palmer, 2002](#)), which are not domain-specific.

Note that our proposed form of structured representations not necessarily allows directly solving informational queries that require explicit reasoning and/or state tracking (e.g., “Where are the tomatoes after step 5?”). We however pose that properly detecting the various entities (e.g., ingredients and their derivations) and the actions that are executed on them (as described by verbs), with the appropriate coreference and zero anaphora resolution, would enable constructing a graph that facilitates such tracking. Thus, while our proposed representation based on the idea of joint entity and relation extraction ([Bekoulis et al., 2018](#)), provides useful input for it, such explicit state tracking and representation (e.g., as in ProPara, [Dalvi et al., 2018](#)) is left out of scope here.

In summary, this paper reports on our work-in-progress and makes two main contributions. First, we present our newly annotated Recipe Instruction Semantic Corpus (RISeC) dataset (Section 3), following the frame-semantic representation of PropBank ([Kingsbury and Palmer, 2002](#)). Since

PropBank is domain-agnostic, the approach should be largely generalizable<sup>1</sup> to other procedural text settings. Second, we introduce a baseline framework (Section 4) to solve (i) entity recognition (ingredients, tools and actions), (ii) relation extraction (ingredients and tools linked to the actions), (iii) zero anaphora identification. Experimental evaluation thereof on RISeC is provided (Section 5).

## 2 Related work

From the perspective of structured representation, Tasse and Smith (2008) define the Minimal Instruction Language for the Kitchen (MILK), which is based on first-order logic to describe the evolution of ingredients throughout a recipe, and use it for annotation in the CURD dataset. Building on this effort, Jermsurawong and Habash (2015) extend CURD toward ingredient-instruction dependency tree parsing in SIMMR: they present an ingredient-instruction dependency tree representation of the recipe, but do not model instruction semantics. This contrasts with Maeta et al. (2015), who propose a pipeline framework for information extraction on Japanese recipes from the the recipe flow graph (r-FG) dataset (Mori et al., 2014). Maeta et al. use word segmentation, named entity recognition and syntactic analysis to extract predicate-argument structures and build a recipe flow graph that is conceptually similar to a SIMMR tree. Their work is conceptually closest to ours, in that they propose a chain of NLP subtasks (but we do not need word boundary identification in our English corpus). Yet, we build on a more elaborate and generic semantic relation scheme, PropBank (Kingsbury and Palmer, 2002). Further, methodologically we adopt neural network models as opposed to their logistic regression for NER and a maximum spanning tree (MST) parser for the relations (i.e., graph arcs). Tracking state changes is another key to understanding recipe language. Bosselut et al. (2018) predict the dynamics of action and entity attributes in recipes by employing a recurrent memory network. Their work includes sentence generation, but does not address the zero anaphora problem (see further) directly.

Besides recipes, other works focus on different procedural tasks. The ProPara<sup>2</sup> project aims at

<sup>1</sup>While some of our entity types are specific to the cooking domain (e.g., “food”, “temperature”), the relations that link action verbs to them are not (cf. PropBank).

<sup>2</sup><http://data.allenai.org/propara>

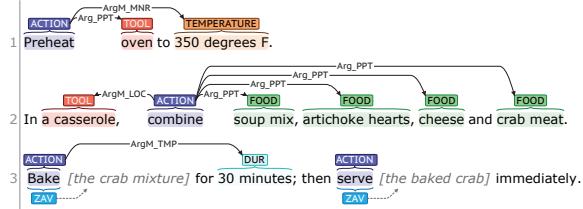


Figure 1: An annotated recipe. The fragments between brackets are manually added anaphora descriptions.

comprehending scientific processes and tracking the status of entities in them: Dalvi et al. (2018) focus on tracking entity locations (as well as their creation/destruction) using a specific matrix state representation (with a row per step, a column per entity). The proposed models however do not incorporate entity recognition and are specifically filling the chosen state representation. In our work, we rather stick to a more “basic” understanding, which is broader in scope than location tracking. In terms of datasets beyond the recipe domain, the work of Mysore et al. (2019) is noteworthy: it focuses on material synthesis and annotates domain-specific entities (materials, operations, conditions, etc.) and relations. The latter in our case are rather domain-agnostic (using PropBank).

## 3 The RISeC Dataset

The following paragraphs describe our dataset and the annotations underlying the presented extraction task<sup>3</sup>.

### 3.1 Dataset Collection

Recipes in our RISeC dataset are those from the SIMMR dataset.<sup>4</sup> Unlike SIMMR, we only use the instruction text of each recipe, and rather detect ingredients (as well as derived entities) from the text itself. We annotate the dataset using BRAT (Stenetorp et al., 2012), which eventually creates a directed acyclic graph where (i) vertices are *entities* (text spans) such as ingredients, tools, actions, intermediate products, and (ii) edges denote *relations* between entity spans. An example of our annotation is given in Fig. 1. Three expert annotators are involved in this task, who were in close communication during the entire annotation process to maximize annotation consistency.

<sup>3</sup>The annotated data is available for research at <https://github.com/YiweiJiang2015/RISeC>

<sup>4</sup><https://camel.abudhabi.nyu.edu/simmr/>

## 3.2 Annotation Structure

### Entity Types

Action: Most verbs, their present/past participles and verb phrases fall in this category. In addition to the Action label, specific verbs also carry a Zero Anaphora Verb (ZAV) label (see further).

Food: Ingredients, spices (salt, sugar, etc.), intermediate products (e.g., “the meat mixture”). If a sequence of ingredients is involved in an action, we label each of them individually, as in Fig. 1.

Tool: Appliances (e.g., oven), recipients (e.g., bowl), utensils (e.g., fork) used to perform an action involved in the cooking process.

Duration: Time interval for which an action lasts (e.g., ‘20 minutes’, ‘half an hour’).

Temperature: E.g., “400 degrees F”.

Other: This label is used for entities that cannot be attributed to any entity label above.

Further, we also annotate subclauses that provide information on certain actions as “entities”. Thus, we abuse entity labeling to indicate them and thus limit their annotation to shallow parsing:

Condition\_Clause: Sub-clauses led by conjunctions like “until”, “till”, “when”, “before”, usually expressing timing.

Purpose\_Clause: Infinitives and sub-clauses led by for example “so that”, “to make sure that”.

### Relation Types

Following the methodology of PropBank, we define a set of relations for the semantic roles in recipe instructions. These relations have the verb as origin and link an action to its arguments ( $\text{Arg}_*$ ) or modifiers ( $\text{ArgM}_*$ ). For details on their meanings, see PropBank’s annotation guidelines (Babko-Malaya, 2005). However, to make the annotating schema self-consistent and adaptive to the cooking domain, we create (or extend) verb frames that are not (yet) included by PropBank.

E.g., for the verb phrase “beat in”, we borrow the argument structure from its main verb, i.e., “beat”.

$\text{Arg\_PPT}$ : Participant, used for the argument which undergoes a change of state or is being affected by an action.

$\text{Arg\_GOL}$ : Goal, destination where an action ends.

$\text{Arg\_DIR}$ : Direction, the source where an action starts from. E.g., “Remove the pan from oven to a rack” where “oven” is  $\text{Arg\_DIR}$  of the action “remove”.

$\text{Arg\_PRD}$ : Predicate, used for the end product of an action. E.g., “Roll the cool dough into 3-inch ball” where the dough is transformed into “3-inch

balls”,  $\text{Arg\_PRD}$  of the action “roll”.

$\text{Arg\_PAG}$ : Agent, the subject that performs an action.

$\text{ArgM\_MNR}$ : Manner, describing how or in what condition we execute an action. E.g., in “Preheat the oven at 340 degrees”, the relation  $\text{ArgM\_MNR}$  links Action “preheat” to Temperature “340 degrees F”.

$\text{ArgM\_LOC}$ : Location where an action takes place. This notion is not restricted to physical locations, but abstract locations are being marked as  $\text{ArgM\_LOC}$  as well. E.g., in “Beat 2 eggs in the flour”,  $\text{ArgM\_LOC}$  links Action “beat” to Food “the flour”

$\text{ArgM\_TMP}$ : Temporal relation between action and timing nodes (Duration, Condition\_clause).

$\text{ArgM\_PRP}$ : Purpose relation between action and purpose clause nodes.

$\text{ArgM\_INT}$ : Instrument, e.g., the utensil to accomplish the action.

$\text{ArgM\_SIM}$ : Simultaneous, linking two actions performed at the same time. E.g., in “Broil the lamb, moving pan so entire surface browns evenly”,  $\text{ArgM\_SIM}$  links “broil” to “moving”.

### Zero Anaphora Rephrasing

Zero anaphora is the phenomenon of implicit, unmentioned references to earlier concepts. Figure 1 gives two examples where explicit anaphors are manually added inside the brackets. The last sentence in Fig. 1 would be ungrammatical without the unmentioned “the crab mixture” and “the baked crab”. In our annotations, we annotated 1,526 Zero Anaphora Verbs with candidate expressions for the zero anaphora, providing at least two alternatives: a succinct noun, as well as a more detailed noun phrase.

## 4 Model

We focus on two tasks: (1) joint entity recognition, relation extraction and zero anaphora identification, and (2) zero anaphora description generation. Next we present our models for each.

### 4.1 Entity recognition, relation extraction & zero anaphora identification

We use a span-based model, taking the input sequence of words as input, and passing it through 4 components: (i) word representation, (ii) span representation, (iii) entity recognition, and (iv) relation identification.

*Word Representation*: We use a BiLSTM as the

base encoder. The inputs are vector representations of the sentence tokens obtained by concatenating pre-trained GLoVe embeddings (Pennington et al., 2014) and character representations (using a CNN, ReLU and max pooling, as proposed by dos Santos and Guimaraes, 2015). Further, we also experimented with pre-trained BERT models (Devlin et al., 2019) instead of Glove embeddings.

*Span Representation:* We enumerate all possible word spans from the input sentence and concatenate the aforementioned BiLSTM ( $h_{left}, h_{right}$ ) encoder outputs at first ( $f$ ) and last ( $l$ ) end-point tokens of each span, together with its length ( $e_{len}$ ) to obtain a span representation ( $s_i = (h_{left,f}, h_{right,f}, h_{left,l}, h_{right,l}, e_{len})$ ).

*Entity Recognition & Zero Anaphora Verb Identification:* We pass the selected span representations  $s_i$  through a feed-forward neural network (FFNN) yielding per-class scores for predicting entity types as well as binary Zero Anaphora Verb labels (with  $k$  entity classes, the FFNN thus has  $k + 1$  outputs).

*Relation Identification:* The concatenation of two span representations ( $s_i, s_j$ ) is passed through another FFNN to derive per-class relation scores. Since this is quadratic, we only pass the top 20% highest scored spans to the Relation FFNN: every span pair ( $s_i, s_j$ ) is first passed through a pruning FFNN, and only its top-scored pairs are pushed through the Relation FFNN.

*Training:* For each recipe instance, the objective is to optimize the weighted sum of the negative log likelihood of span representation, entity classification and relation identification. We use Adam to optimize the model with learning rate 0.001.

## 4.2 Zero anaphora description generation

For the generation task, we build a baseline model corresponding to the sequence-to-sequence architecture used in Bahdanau et al. (2015). The input is the entire recipe, which we pass to an LSTM encoder taking the pre-trained GloVe embedding, concatenated with a binary label indicating whether it is a zero anaphora verb (ZAV), and (optionally) an entity type embedding if the token is of a given type. Since usually the target description that the decoder needs to generate is much shorter than the full recipe, we adopt bilinear attention (Luong et al., 2015). The model is trained to minimize the negative log likelihood of

	Glove	Bert <sub>base</sub>	Bert <sub>large</sub>
Entity	89.8	91.7	92.6
Zero Anaphora Verb	89.1	89.0	89.8
Relation	65.5	67.1	67.5

Table 1: Micro-F1 scores of models with Glove, Bert<sub>base</sub> and Bert<sub>large</sub> on the test set.

	Full	Test set		
	Count	Prec.	Recall	F1
Food	3,232	92.5	95.9	94.2
Action	3,061	96.6	97.4	97.0
Tool	1,138	92.9	86.8	89.8
Condition clause	487	93.0	71.1	80.5
Duration	411	85.7	87.4	86.5
Temperature	381	87.4	89.3	88.4
Other	270	54.2	34.7	41.9
Purpose clause	147	78.0	59.2	67.2

Table 2: Entity counts in full dataset and extraction results with Bert<sub>large</sub> on test set.

an emitted token given the full input and predicted tokens.

## 5 Experiments and results

We split our RISeC dataset into 50% training, 20% development and 30% test sets, using the same splits as SIMMR (Jerm surawong and Habash, 2015). We tune hyperparameters on the development set. Reported performance metrics are obtained on the test set.

In general, our span-based model shows good performance in the extraction task, as shown in Table 1. We obtain micro-F1 scores for the joint entity, zero anaphora verbs and relation identification tasks of respectively 89.8, 89.1 and 65.5 when using Glove word embeddings. With Bert<sub>large</sub> word encodings, performance consistently improves by 2.8, 0.7 and 2.0 percentage points respectively, indicating the applicability of the general linguistic knowledge from Bert on a cooking-domain task.

Individual entity and relation type performance is reported in Tables 2–3. As expected, Table 2 shows that entity F1 scores are positively correlated with the occurrence frequency, except for Duration and Temperature, of which the fixed pattern is easy to learn. The high precision and recall of important entities like Food and Action shows promising potential of our model for downstream applications like a question answering system in smart kitchen settings. The F1

		Full	Test set		
		Count	Prec.	Recall	F1
Argument Relations	Arg_PPT	3,196	94.1	69.3	79.8
	Arg_GOL	557	79.6	35.8	49.1
	Arg_DIR	91	93.9	34.5	50.4
	Arg_PRD	74	77.8	27.4	40.0
	Arg_PAG	25	0.0	0.0	0.0
Modifier Relations	ArgM_TMP	884	91.7	33.2	48.7
	ArgM_LOC	515	87.8	49.7	63.3
	ArgM_MNR	432	86.7	35.6	50.1
	ArgM_PRP	137	85.2	9.1	15.8
	ArgM_SIM	92	66.7	11.1	18.6
	ArgM_INT	73	77.4	20.3	31.8

Table 3: Relation counts in full dataset and extraction results with Bert<sub>large</sub> on test set.

scores of relation predictions in Table 3 show that the imbalanced distribution of relation types causes detection of several relations to be difficult, e.g., the low recall rates for Arg\_PAG and ArgM\_PRP. Future work should address this, e.g., using a larger dataset (or pretraining on non-recipe corpora).

While the detection of zero anaphora verbs (ZAV) performs well, our Seq2seq based description generation largely failed, with very low performance and oftentimes outputting the same descriptions (e.g., “mixture” or “chicken”). In hindsight, given the limited dataset size (order of 1.5k ZAV occurrences in the full dataset) and the typically large training dataset needed for seq2seq models, this is not entirely unexpected. Further work on this task is clearly required.

## 6 Conclusion and Future Work

This paper introduced RISeC, a dataset for extracting structural information and resolving zero anaphora from unstructured recipes. The corpus consists of 260 recipes from SIMMR and provides semantic graph annotations of (i) recipe-related entities, (ii) generic verb relations (from PropBank) connecting these entities, (iii) zero anaphora verbs having implicit arguments, and (iv) textual descriptions of those implicit arguments. We reported on our work-in-progress with two baseline models using our corpus: (i) a neural span-based model extracting entities, zero anaphora verbs and relations, and (ii) a sequence-to-sequence attention model generating noun phrases for zero anaphora verbs.

We plan to continue working in this direction, making the dataset larger and more fine-grained, and especially, to investigate how it

can be leveraged for human-machine interaction experiments.

## Acknowledgments

The first author was supported by *China Scholarship Council* (201806020194). This research received funding from the Flemish Government under the “Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen” programme. We would like to thank anonymous reviewers who helped to improve the draft.

## References

- Olga Babko-Malaya. 2005. Propbank annotation guidelines.
- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the 2015 International Conference on Learning Representations (ICLR 2015)*, San Diego, CA, USA.
- Ioannis Bekoulis, Johannes Deleu, Thomas Demeester, and Chris Develder. 2018. Joint entity recognition and relation extraction as a multi-head selection problem. *Expert Systems with Applications*, 114:34–45.
- Antoine Bosselut, Omer Levy, Ari Holtzman, Corin Ennis, Dieter Fox, and Yejin Choi. 2018. Simulating action dynamics with neural process networks. *ArXiv preprint arXiv:1711.05313*.
- Bhavana Dalvi, Lifu Huang, Niket Tandon, Wentaui Yih, and Peter Clark. 2018. Tracking state changes in procedural text: a challenge dataset and models for process paragraph comprehension. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT 2018)*, pages 1595–1604, New Orleans, Louisiana.
- J. Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.
- Jermsak Jermsurawong and Nizar Habash. 2015. Predicting the structure of cooking recipes. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 781–786, Lisbon, Portugal. Association for Computational Linguistics.
- Paul Kingsbury and Martha Palmer. 2002. From TreeBank to PropBank. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC 2002)*, pages 1989–1993, Las Palmas, Canary Islands, Spain.

- Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. **Effective approaches to attention-based neural machine translation**. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP 2015)*, pages 1412–1421, Lisbon, Portugal.
- Hirokuni Maeta, Tetsuro Sasada, and Shinsuke Mori. 2015. **A framework for procedural text understanding**. In *Proceedings of the 14th International Conference on Parsing Technologies (IWPT 2015)*, pages 50–60, Bilbao, Spain.
- Shinsuke Mori, Hirokuni Maeta, Yoko Yamakata, and Tetsuro Sasada. 2014. **Flow graph corpus from recipe texts**. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC 2014)*, pages 2370–2377, Reykjavik, Iceland.
- Sheshera Mysore, Zachary Jensen, Edward Kim, Kevin Huang, Haw-Shiuan Chang, Emma Strubell, Jeffrey Flanigan, Andrew McCallum, and Elsa Olivetti. 2019. **The materials science procedural text corpus: Annotating materials synthesis procedures with shallow semantic structures**. In *Proceedings of the 13th Linguistic Annotation Workshop*, pages 56–64, Florence, Italy.
- Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. **GloVe: Global vectors for word representation**. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar.
- Cícero dos Santos and Victor Guimarães. 2015. **Boosting named entity recognition with neural character embeddings**. In *Proceedings of the Fifth Named Entity Workshop*, pages 25–33, Beijing, China.
- Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun’ichi Tsujii. 2012. **Brat: a web-based tool for NLP-assisted text annotation**. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2012)*, pages 102–107, Avignon, France.
- Dan Tasse and Noah A Smith. 2008. **SOUR CREAM: Toward semantic processing of recipes**. *Carnegie Mellon University, Pittsburgh, Tech. Rep. CMU-LTI-08-005*.

# Stronger Baselines for Grammatical Error Correction Using a Pretrained Encoder–Decoder Model

Satoru Katsumata\* and Mamoru Komachi

Tokyo Metropolitan University

satoru.katsumata@retrieva.jp, komachi@tmu.ac.jp

## Abstract

Studies on grammatical error correction (GEC) have reported the effectiveness of pre-training a Seq2Seq model with a large amount of pseudodata. However, this approach requires time-consuming pretraining for GEC because of the size of the pseudodata. In this study, we explore the utility of bidirectional and auto-regressive transformers (BART) as a generic pretrained encoder–decoder model for GEC. With the use of this generic pretrained model for GEC, the time-consuming pretraining can be eliminated. We find that monolingual and multilingual BART models achieve high performance in GEC, with one of the results being comparable to the current strong results in English GEC. Our implementations are publicly available at GitHub<sup>1</sup>.

## 1 Introduction

Grammatical error correction (GEC) is the automatic correction of grammatical and other language-related errors in text. Most works regard this task as a translation task and use encoder–decoder (Enc–Dec) architectures to convert ungrammatical sentences to grammatical ones. This Enc–Dec approach often does not require linguistic knowledge of the target language. Strong Enc–Dec models for GEC are pretrained with a large amount of artificially generated data, commonly referred to as ‘pseudodata’, that is created by introducing artificial error to a monolingual corpus. Hereafter, pretraining using pseudodata aimed at the GEC task is referred to as **task-oriented** pretraining (Kiyono et al., 2019; Grundkiewicz et al., 2019; Náplava and Straka, 2019; Kaneko et al., 2020). For example, Kiyono et al. (2019) generated a pseudo corpus using back-translation and

achieved strong results for English GEC. Náplava and Straka (2019) generated a pseudo corpus by introducing artificial errors into monolingual corpora and achieved the best scores for GEC in several languages by adopting the methods proposed by Grundkiewicz et al. (2019).

These task-oriented pretraining approaches require extensive use of a pseudo-parallel corpus. Specifically, Grundkiewicz et al. (2019) used 100M ungrammatical and grammatical sentence pairs, while Kiyono et al. (2019) and Kaneko et al. (2020) used 70M sentence pairs, which required time-consuming pretraining of GEC models using the pseudo corpus.

In this study, we determined the effectiveness of publicly available pretrained Enc–Dec models for GEC. Specifically, we investigated pretrained models without the need for pseudodata. We explored a pretrained model proposed by Lewis et al. (2020) called bidirectional and auto-regressive transformers (BART). Liu et al. (2020) also proposed multilingual BART. These models were pretrained by predicting the original sequence, given a masked and shuffled sentence. The motivation for using these models for GEC was that it achieved strong results for several text generation tasks, such as summarization; we refer to it as a **generic** pretrained model.

We used generic pretrained BART models to compare with GEC models using a pseudo-corpus approach (Kiyono et al., 2019; Kaneko et al., 2020; Náplava and Straka, 2019). We conducted GEC experiments for four languages: English, German, Czech, and Russian. The Enc–Dec model based on BART achieved results comparable with those of current strong Enc–Dec models for English GEC. The multilingual model also showed high performance in other languages, despite only requiring fine-tuning. These results suggest that BART can be used as a simple baseline

\*Currently working at Retrieva, Inc.

<sup>1</sup><https://github.com/Katsumata420/generic-pretrained-GEC>

for GEC.

## 2 Previous Work

The Enc-Dec approach for GEC often uses the task-oriented pretraining strategy. For example, Zhao et al. (2019) and Grundkiewicz et al. (2019) reported that pretraining of the Enc-Dec model using a pseudo corpus is effective for the GEC task. In particular, they introduced word- and character-level errors into a sentence in monolingual corpora. They developed a confusion set derived from a spellchecker and randomly replaced a word in a sentence. They also randomly deleted a word, inserted a random word, and swapped a word with an adjacent word. They performed these same operations, i.e., replacing, deleting, inserting, and swapping, for characters. The pseudo corpus made by the above methods consisted of 100M training samples. Our study aims to investigate whether the generic pretrained models are effective for GEC, because pretraining with such a large corpus is time-consuming.

Náplava and Straka (2019) adopted Grundkiewicz et al. (2019)'s method for several languages, including German, Czech, and Russian. They trained a Transformer (Vaswani et al., 2017) with pseudo corpora (10M sentence pairs), and achieved current state-of-the-art (SOTA) results for German, Czech, and Russian GEC. We compared their results with those of the generic pretrained model to confirm whether the model was effective for GEC in several languages.

Kiyono et al. (2019) explored the generation of a pseudo corpus by introducing random errors or using back-translation. They reported that a task-oriented pretraining with back-translation data and character errors is better than that with pseudodata based on random errors. Kaneko et al. (2020) combined Kiyono et al. (2019)'s pretraining approach with BERT (Devlin et al., 2019) and improved Kiyono et al. (2019)'s results. Specifically, Kaneko et al. (2020) fine-tuned BERT with a grammatical error detection task. The fine-tuned BERT outputs for each token were combined with the original tokens as a GEC input. Their study is similar to our research in that both studies use publicly available generic pretrained models to perform GEC. The difference between these studies is that Kaneko et al. (2020) used the architecture of the pretrained model as an encoder. Therefore, their method still requires pretraining with a large

amount of pseudodata.

The current SOTA approach for English GEC uses the sequence tagging model proposed by Omelianchuk et al. (2020). They designed token-level transformations to map input tokens to target corrections to produce training data. The sequence tagging model then predicts the transformation corresponding to the input token. We do not attempt to make a comparison with this approach, as the purpose of our study is to create a strong GEC model without using pseudodata or linguistic knowledge.

## 3 Generic Pretrained Model

BART (Lewis et al., 2020) is pretrained by predicting an original sequence, given a masked and shuffled sequence using a Transformer. They introduced masked tokens with various lengths based on the Poisson distribution, inspired by SpanBERT (Joshi et al., 2020), at multiple positions. BART is pretrained with large monolingual corpora (160 GB), including news, books, stories, and web-text domains. This model achieved strong results in several generation tasks; thus, it is regarded as a generic model.

They released pretrained models using English monolingual corpora for several tasks, including summarization, which we used for English GEC. Liu et al. (2020) proposed multilingual BART (mBART) for a machine translation task, which we used for GEC of several languages. The latter model was trained using monolingual corpora for 25 languages simultaneously. They used a special token for representing the language of a sentence. For example, they added <de\_DE> and <rue\_RU> into the initial token of the encoder and decoder for De-Ru translation. To fine-tune mBART for German, Czech, and Russian GEC, we set the target language for the special token referring to that language.

## 4 Experiment

### 4.1 Settings

**Common Settings.** As presented in Table 1, we used learner corpora, including BEA<sup>2</sup> (Bryant et al., 2019; Granger, 1998; Mizumoto et al., 2011; Tajiri et al., 2012; Yannakoudakis et al., 2011; Dahlmeier et al., 2013), JFLEG (Napoles et al., 2017), and CoNLL-14 (Ng et al., 2014) data for

<sup>2</sup>BEA corpus is made of several corpora. Details can be found in Bryant et al. (2019).

lang	Corpus	Train	Dev	Test
En	BEA	1,157,370	4,384	4,477
	JFLEG	-	-	747
	CoNLL-2014	-	-	1,312
De	Falko+MERLIN	19,237	2,503	2,337
Cz	AKCES-GEC	42,210	2,485	2,676
Ru	RULEC-GEC	4,980	2,500	5,000

Table 1: Data statistics.

English; Falko+MERLIN data (Boyd et al., 2014) for German; AKCES-GEC (Náplava and Straka, 2019) for Czech; and RULEC-GEC (Rozovskaya and Roth, 2019) for Russian.

Our models were fine-tuned using a single GPU (NVIDIA TITAN RTX), and our implementations were based on publicly available code<sup>3</sup>. We used the hyperparameters provided in some previous works (Lewis et al., 2020; Liu et al., 2020), unless otherwise noted.

The scores excluding the ensemble method were averaged in five fine-tuned experiments with random seeds.

**English.** Our setting for the English datasets was almost the same as that of Kiyono et al. (2019). We extracted the training data from BEA-train for English GEC. Similar to Kiyono et al. (2019), we did not use the unchanged sentences in the source and target sides; thus, the training data consisted of 561,525 sentences. We used BEA-dev to determine the best model.

We trained the BART-based models by using `bart.large`. This model was proposed for the summarization task, which required some constraints in inference to ensure appropriate outputs; however, we did not impose any constraints because our task was different. We applied byte pair encoding (BPE) (Sennrich et al., 2016) to the training data for the BART-based model by using the BPE model of Lewis et al. (2020).

We used the  $M^2$  scorer (Dahlmeier and Ng, 2012) and GLEU (Napoles et al., 2015) for CoNLL-14 and JFLEG, respectively, and used the ERRANT scorer (Bryant et al., 2017) for BEA-test. We compared these scores with strong results (Kiyono et al., 2019; Kaneko et al., 2020).

**German, Czech, and Russian.** The dataset settings in this study were almost the same as those

used by Náplava and Straka (2019) for each language. We used official training data and decided the best model by using the development data.

In addition, we trained the mBART-based models for German, Czech, and Russian GEC. We used `mbart.cc25` for the mBART-based models. For the mBART-based model, we followed Liu et al. (2020); we detokenized<sup>4</sup> the GEC training data for the mBART-based model and applied SentencePiece (Kudo and Richardson, 2018) with the SentencePiece model shared by Liu et al. (2020). Using this preprocessing, the input sentence may not represent grammatical information, compared with the sentence tokenized using a morphological analysis tool and subword tokenizer. However, what preprocessing is appropriate for GEC is beyond this paper’s scope and will be treated as future work. For evaluation, we tokenized the outputs after recovering the subwords. Then, we used a spaCy-based<sup>5</sup> tokenizer for German<sup>6</sup> and Russian<sup>7</sup>, and the MorphoDiTa tokenizer<sup>8</sup> for Czech.

Moreover, the  $M^2$  scorer was used for each language. We compared these scores with the current SOTA results (Náplava and Straka, 2019).

## 4.2 Results

**English.** Table 2 presents the results of the English GEC task. When using a single model, the BART-based model is better than the model proposed by Kiyono et al. (2019), and the results are comparable to those reported by Kaneko et al. (2020) in terms of CoNLL-14 and BEA-test. Kiyono et al. (2019) and Kaneko et al. (2020) incorporated several techniques to improve the accuracy of GEC. To compare these models, we experimented with an ensemble of five models. Our ensemble model was slightly better than our single model, but worse than the ensemble models by Kiyono et al. (2019) and Kaneko et al. (2020). The BART-based model along with the ensemble model achieved results comparable to current strong results despite only requiring fine-tuning of the BART model. We believe that the reason for the ineffectiveness of the ensemble method is that the five models are not significantly different as the

<sup>4</sup>We used `detokenizer.perl` in the Moses script (Koehn et al., 2007).

<sup>5</sup><https://spacy.io>

<sup>6</sup>We used the built-in de model.

<sup>7</sup>[https://github.com/aatimofeev/spacy\\_russian\\_tokenizer](https://github.com/aatimofeev/spacy_russian_tokenizer)

<sup>8</sup><https://github.com/ufal/morphodita>

<sup>3</sup>BART, mBART: <https://github.com/pytorch/fairseq>

	CoNLL-14 ( $M^2$ )			JFLEG	BEA-test		
	P	R	$F_{0.5}$		P	R	$F_{0.5}$
Kiyono et al. (2019)	67.9/ <u>73.3</u>	44.1/44.2	61.3/64.7	59.7/61.2	65.5/ <u>74.7</u>	59.4/56.7	64.2/70.2
Kaneko et al. (2020)	69.2/ <u>72.6</u>	<b>45.6/46.4</b>	<b>62.6/65.2</b>	<b>61.3/62.0</b>	67.1/72.3	<b>60.1/61.4</b>	<b>65.6/69.8</b>
BART-based	<b>69.3/69.9</b>	45.0/45.1	<b>62.6/63.0</b>	57.3/57.2	<b>68.3/68.8</b>	57.1/57.1	<b>65.6/66.1</b>

Table 2: English GEC results. Left and right scores represent single and ensemble model results, respectively. Bold scores represent the best score in the single models, and underlined scores represent the best overall score.

		P	R	$F_{0.5}$
De	Náplava and Straka (2019)	78.21	59.94	73.31
	mBART-based	73.97	53.98	68.86
Cz	Náplava and Straka (2019)	83.75	68.48	80.17
	mBART-based	78.48	58.70	73.52
Ru	Náplava and Straka (2019)	63.26	27.50	50.20
	mBART-based	32.13	4.99	15.38
	with pseudo corpus	53.50	26.35	44.36

Table 3: German, Czech, and Russian GEC results. These models are not an ensemble of multiple models.

initial weights are the same as those of the BART model, and seeds only affect minor changes, such as training data order, and so on.

**German, Czech, and Russian.** Table 3 presents the results for German, Czech, and Russian GEC.

In the German GEC task, the mBART-based model achieves 4.45  $F_{0.5}$  points lower than the model by Náplava and Straka (2019). This may be because Náplava and Straka (2019) pretrains the GEC model with only the target language, whereas mBART is pretrained with 25 languages, resulting in the information of other languages being included as noise.

In the Czech GEC task, the mBART-based model achieves 6.65  $F_{0.5}$  points lower than the model by Náplava and Straka (2019). Similar to the case of the German GEC results, we suppose that mBART includes noisy information.

Considering Russian GEC, the mBART-based model shows much lower scores than Náplava and Straka (2019)'s model. This may be because the training data for Russian GEC are scarce compared to those of German or Czech. To investigate the effect of corpus size, we additionally trained the mBART model with a 10M pseudo corpus, using the method proposed by Grundkiewicz et al. (2019), and fine-tuned it with the learner corpus to compensate for the low-resource scenario. The results presented in Table 3 support our hypothesis.

Error Type	Kaneko et al. (2020)			BART-based		
	P	R	$F_{0.5}$	P	R	$F_{0.5}$
PUNCT	74.1	52.7	68.5	79.2	59.0	<b>74.1</b>
DET	73.7	72.9	73.5	76.3	71.1	<b>75.2</b>
PREP	73.4	69.1	<b>72.5</b>	71.2	64.8	69.9
ORTH	86.9	62.9	<b>80.8</b>	84.2	52.9	75.3
SPELL	83.1	79.5	<b>82.3</b>	84.7	55.2	76.5

Table 4: BEA-test scores for the top five error types, except for OTHER. Kaneko et al. (2020) and BART-based are ensemble models. Bold scores represent the best score for each error type.

## 5 Discussion

**BART as a simple baseline model.** According to the German and Czech GEC results, the mBART-based model, in which we only fine-tuned the pretrained mBART model, achieves comparable scores with SOTA models. In other words, mBART-based models are considered to show sufficiently high performance for several languages without using a pseudo corpus. These results indicate that the mBART-based model can be used as a simple GEC baseline for several languages.

**Performance comparison for each error type.** We compare the BART-based model with Kaneko et al. (2020)'s model for common error types using a generic pretrained model. Table 4 presents the results for the top five error types in BEA-test. According to these results, BART-based is superior to Kaneko et al. (2020) in PUNCT and DET errors; in particular, PUNCT is 5.6  $F_{0.5}$  points better. BART is pretrained to correct the shuffled and masked sequence, so that this model learns to place punctuation adequately. In contrast, Kaneko et al. (2020) uses an encoder that is not pretrained with correcting shuffled sequences.

Conversely, Kaneko et al. (2020) report better results for other errors, except for DET. Regarding ORTH and SPELL, their model is more than 5  $F_{0.5}$  points better than the BART-based one. It is difficult for the BART-based model to cor-

rect these errors because BART uses shuffled and masked sequences as noise in pretraining; not using character-level errors. Kaneko et al. (2020) introduce character errors into a pseudo corpus as task-oriented Enc-Dec pretraining; this is the reason why the BART-based model is inferior to Kaneko et al. (2020) in these errors.

## 6 Conclusion

We introduced a generic pretrained Enc-Dec model, BART, for GEC. The experimental results indicated that BART better initialized the Enc-Dec model parameters. The fine-tuned BART achieved remarkable results, which were comparable to the current strong results in English GEC. Indeed, the monolingual BART seems to be more effective for GEC than the model with a multilingual setting. However, although it is not as good as SOTA, fine-tuned mBART exhibited high performance in other languages. This implies that BART is a simple baseline model for pretraining GEC methods because it only requires fine-tuning as training.

## Acknowledgements

We thank the anonymous reviewers for their insightful comments. This work has been partly supported by the programs of the Grant-in-Aid for Scientific Research from the Japan Society for the Promotion of Science (JSPS KAKENHI) Grant Numbers 19K12099 and 19KK0286.

## References

- Adriane Boyd, Jirka Hana, Lionel Nicolas, Detmar Meurers, Katrin Wisniewski, Andrea Abel, Karin Schöne, Barbora Štindlová, and Chiara Vettori. 2014. The MERLIN corpus: Learner language and the CEFR. In *Proc. of LREC*, pages 1281–1288.
- Christopher Bryant, Mariano Felice, and Ted Briscoe. 2017. Automatic annotation and evaluation of error types for grammatical error correction. In *Proc. of ACL*, pages 793–805.
- Christopher Bryant, Mariano Felice, ¥Oistein E. Andersen, and Ted Briscoe. 2019. The BEA-2019 shared task on grammatical error correction. In *Proc. of BEA*, pages 52–75.
- Daniel Dahlmeier and Hwee Tou Ng. 2012. Better evaluation for grammatical error correction. In *Proc. of NAACL-HLT*, pages 568–572.
- Daniel Dahlmeier, Hwee Tou Ng, and Siew Mei Wu. 2013. Building a large annotated corpus of learner English: The NUS corpus of learner English. In *Proc. of BEA*, pages 22–31.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of NAACL-HLT*, pages 4171–4186.
- Sylviane Granger. 1998. The computer learner corpus: A versatile new source of data for SLA research. In Sylviane Granger, editor, *Learner English on Computer*, pages 3–18. Addison Wesley Longman.
- Roman Grundkiewicz, Marcin Junczys-Dowmunt, and Kenneth Heafield. 2019. Neural grammatical error correction systems with unsupervised pre-training on synthetic data. In *Proc. of BEA*, pages 252–263.
- Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2020. Span-BERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.
- Masahiro Kaneko, Masato Mita, Shun Kiyono, Jun Suzuki, and Kentaro Inui. 2020. Can encoder-decoder models benefit from pre-trained language representation in grammatical error correction? In *Proc. of ACL*.
- Shun Kiyono, Jun Suzuki, Masato Mita, Tomoya Mizumoto, and Kentaro Inui. 2019. An empirical study of incorporating pseudo data into grammatical error correction. In *Proc. of EMNLP-IJCNLP*, pages 1236–1242.
- Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proc. of ACL Demo Sessions*, pages 177–180.
- Taku Kudo and John Richardson. 2018. Sentence-Piece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proc. of EMNLP: System Demonstrations*, pages 66–71.
- Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proc. of ACL*.
- Yinhan Liu, Jiatao Gu, Naman Goyal, Xiongmin Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *ArXiv*, abs/2001.08210.

Tomoya Mizumoto, Mamoru Komachi, Masaaki Nagata, and Yuji Matsumoto. 2011. Mining revision log of language learning SNS for automated Japanese error correction of second language learners. In *Proc. of IJCNLP*, pages 147–155.

Jakub Náplava and Milan Straka. 2019. Grammatical error correction in low-resource scenarios. In *Proc. of W-NUT*, pages 346–356.

Courtney Napoles, Keisuke Sakaguchi, Matt Post, and Joel Tetreault. 2015. Ground truth for grammatical error correction metrics. In *Proc. of ACL-IJCNLP*, pages 588–593.

Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. 2017. JFLEG: A fluency corpus and benchmark for grammatical error correction. In *Proc. of EACL*, pages 229–234.

Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. 2014. The CoNLL-2014 shared task on grammatical error correction. In *Proc. of CoNLL Shared Task*, pages 1–14.

Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem Chernodub, and Oleksandr Skurzhanskyi. 2020. GECToR – grammatical error correction: Tag, not rewrite. In *Proc. of BEA*, pages 163–170.

Alla Rozovskaya and Dan Roth. 2019. Grammar error correction in morphologically rich languages: The case of Russian. *Transactions of the Association for Computational Linguistics*, 7:1–17.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proc. of ACL*, pages 1715–1725.

Toshikazu Tajiri, Mamoru Komachi, and Yuji Matsumoto. 2012. Tense and aspect error correction for ESL learners using global context. In *Proc. of ACL*, pages 198–202.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proc. of NeurIPS*, pages 5998–6008.

Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. A new dataset and method for automatically grading ESOL texts. In *Proc. of ACL-HLT*, pages 180–189.

Wei Zhao, Liang Wang, Kewei Shen, Ruoyu Jia, and Jingming Liu. 2019. Improving grammatical error correction via pre-training a copy-augmented architecture with unlabeled data. In *Proc. of NAACL-HLT*, pages 156–165.

# Sina Mandarin Alphabetical Words: A Web-driven Code-mixing Lexical Resource

Rong Xiang<sup>1</sup>, Mingyu Wan<sup>12</sup>, Qi Su<sup>2</sup>, Chu-Ren Huang<sup>1</sup>, Qin Lu<sup>1</sup>

<sup>1</sup>The Hong Kong Polytechnic University, 11 Yuk Choi Road, Hong Kong (China)

csrxiang,csluqin@comp.polyu.edu.hk, churen.huang@polyu.edu.hk

<sup>2</sup>Peking University, 5 Yiheyuan Road, Beijing (China)

wanmy, sukia@pku.edu.cn

## Abstract

Mandarin Alphabetical Word (MAW) is one indispensable component of Modern Chinese that demonstrates unique code-mixing idiosyncrasies influenced by language exchanges. Yet, this interesting phenomenon has not been properly addressed and is mostly excluded from the Chinese language system. This paper addresses the core problem of MAW identification and proposes to construct a large collection of MAWs from Sina Weibo (SMAW) using an automatic web-based technique which includes rule-based identification, informatics-based extraction, as well as Baidu search engine validation. A collection of 16,207 qualified SMAWs are obtained using this technique along with an annotated corpus of more than 200,000 sentences for linguistic research and applicable inquiries.

## 1 Introduction

Mandarin Alphabetic Words (MAWs), also known as lettered words (Liu, 1994) or code-mixing words (Nguyen and Cornips, 2016), are usually formed by Latin, Greek, Arabic alphabets in combination with Chinese characters, e.g. “X-光/X射线”, *X-ray*. Although pure alphabets (e.g. “NBA”) used in Chinese context have also been regarded as MAWs in some previous work (Liu, 1994; Huang and Liu, 2017), they are more like switching-codes that retain the orthography and linguistic behaviors of the original language, instead of showing typical Chinese lexical characteristics. It is noteworthy that MAWs shall be taken as a code-mixing phenomenon instead of code-switching as a MAW is still a Chinese word which is not switched into another language. Therefore, in this work, MAWS refer to the combined type which encodes both alphabet(s) and Chinese character(s) in one word, such as “A型”, *A-type*, “PO主”, *post owner*, and “γ线”, *Gamma Ray*.

It is linguistically-interesting and applicably-significant to investigate MAWs due to two main reasons. First, A MAW maintains part of the Chinese characteristics in morphology, phonology and orthography (e.g. “PK过”, *player killed*, past tense). Meanwhile, it also demonstrates some properties of the foreigner language (e.g. “维生素ing”, *supplementing Vitamin*, progressive)), providing a unique lexical resource for studying morpho-phonological idiosyncrasies of code-mixing words. Second, MAWs serve as an indispensable part of people’s daily vocabulary, especially under the rapid development of social media communication. Yet, being out-liars of the Chinese lexicon, they can cause problems to existing word segmentation/new word extraction tools that are trained on traditional words (Chen and Liu, 1992; Xue and Shen, 2003).

Consider the following example:

E1:	po主也不知道链接被吞了 (The post owner didn't know that the link has been hacked off)
Seg:	po/主/也/不/知道/链接/被/吞/了
Golden Seg:	po主/也/不/知道/链接/被/吞/了

The sentence in E1 (example 1) is segmented using Stanford Parser (Manning et al., 2014) which fails to identify the word “PO主”, *post owner* and breaks it into two parts. The same type of error also occurs in other popular segmentation tools. Although Huang et al. (2007) proposed a radical method of word segmentation to meet the challenge, using a concept of classifying a string of character-boundaries into either word-boundaries or non-word-boundaries, their work did not address the cases of code-mixing words, whose word boundaries can also fall on foreigner alphabets. Some other methods mainly rely on unsupervised methods (Chang and Su, 1997) or simple statistical methods based on N-gram frequencies, with indices of collocation and co-occurrence (Chang

and Su, 1997; Chen and Ma, 2002; Dias, 2003). However, these works are mainly designed for new words of pure Chinese characters, which are not applicable to MAWs.

In this paper, we address the issue of MAW identification and present the construction of the **Sina MAW lexicon (SMAW)** (available at <https://github.com/Christainx/SMAW>) using a fully automatic information extraction technique. The quality of the MAWs (accurateness and inter-rater agreement) are rated by three experts for system evaluation. Compared to previous resources, this dataset provides an unprecedentedly large, balanced, and structured MAWs as well as a MAW annotated corpus. With the availability of a comprehensive MAWs as a valuable Chinese lexical resource as well as corpus resource, it shall benefit many Chinese language processing tasks which need to deal with code-mixing, such as word segmentation and information extraction.

## 2 Related Works

The earliest MAW was probably “X射线/X光”, *X-ray*, which was officially documented in 1903 (Zhang, 2005). For over 60 years, such words had been largely confined to technical and medical domains with very few lexicalized and registered terms in dictionaries. The authoritative *Xiandai Hanyu Cidian/XianHan* (“现代汉语词典”), for instance, initiated a separate section to include 39 MAW entries in 1996. This list has grown rapidly with each subsequent XianHan dictionary edition, reaching 239 entries by the 2012 edition. This in turn generated a flurry of related linguistic studies, which were mainly focused on lexicological and language policy issues (Su and Wu, 2013; Zhang, 2013). Some works have dealt with the emergence of MAWs in light of globalization, placing them in a socio-cultural context (Kozha, 2012; Miao, 2005), and a few are also interested in studying the morpho-lexical status of MAWs (Lun, 2013; Riha and Baker, 2010; Riha, 2010).

In the age of Internet and social media, the scale of MAWs, their extraction methods, and resources of MAWs have changed drastically since the last decade. For example, Zheng et al. (2005) extracted a small set of MAWs with manual validation from the corpus of People’s Daily (Year 2002). Jiang and Dang (2007) extracted 93 MAWs (out of 1,053 new domain-specific terms) using a statistical approach with rule-based validation. Recently, Huang

and Liu (2017) extracted over 1,157 MAWs from both the Sinica Corpus (Chen et al., 1996) and the Chinese Gigaword Corpus (Huang, 2009) based on manually segmented MAWs in the corpora. Although they have extracted 60,000 tokens with alphabetical letters. However, the list mainly includes pure alphabets those are indeed switching codes of other languages. In our study, these pure code-switching words are excluded according to our definition. Their work has established a taxonomy of distributional patterns of alphabetical letters in MAWs and found that typical MAWs follow Chinese modifier-modified (head) morphological rule and the most frequent and productive pattern is alphabetical letter+ mandarin character (AC), such as *type B* in the form of “B型”.

Besides the above investigations, MAWs have not been identified in a systemic and automatic way. The problem of identifying MAWs can be generalized as an issue of new/unknown/out-of-vocabulary word extraction (code-mixing Chinese words in particular) (Chen and Ma, 2002; Zhang et al., 2010). A commonly adopted way of identifying a new word usually rely on word segmentation at the first step and then map the valid MAWs to an existing dictionary. Those not mapped in the dictionary will be identified as new words. This is actually problematic for identifying MAWs (cf. example in Section 1). In addition, previous studies mainly extract MAWs from manually segmented newspapers in pre-1990s (Huang and Liu, 2017). Hence, the resources are domain-constrained and usage-outdated.

## 3 Construction of SMAW

To address the bias in previous works, we propose to collect an MAW list using social-media text commonly commonly available on Sina Weibo platform (Weibo for short, or micro-blogs), a near-natural context. Weibo is one of the most popular social media platform in China with over 400 million active users on monthly basis. This platform becomes the enabler for generating tons of online data, which can serve as a huge Web corpus. The raw dataset crawled from Weibo consists of over 226 million posts (around 20 gigabytes data).

On the other hand, as there are many debates among linguists about the definition of a MAW (Ding et al., 2017; Liu, 1994; Tan et al., 2005; Xue, 2007; Liu, 2002), this work uses a data-driven statistical approach as well as leveraging

on search engine hits to exclude pseudo-MAWs of low-vitality. Details of the methodology are given in the next section.

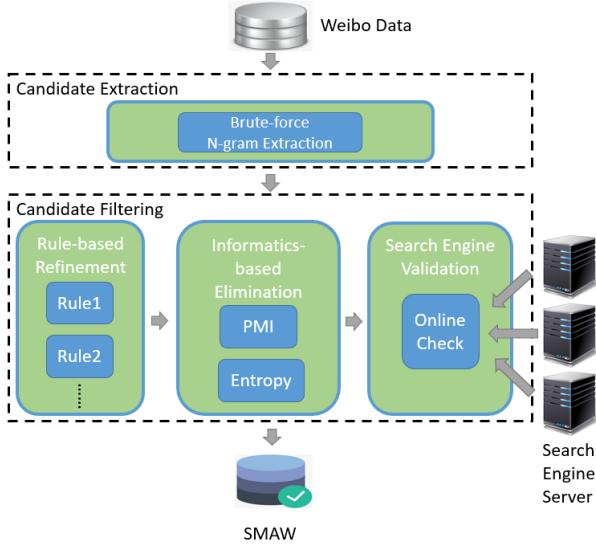


Figure 1: The framework of SMAW construction

Figure 1 depicts the framework of SMAW construction. Collecting the SMAW dataset is carried out through a two stage process: **Candidate Extraction** and **Candidate Filtering**. In our system, **Candidate Extraction** uses an alphabet-anchored brute-force extraction of N-grams tokens which contains both alphabets and Chinese.

To eliminate as many false positive cases as possible, **Candidate Filtering** uses three methods to remove noisy candidates using (1) Rule-based Refinement, (2) Informatics-based Elimination, as well as (3) Search Engine Validation.

In rule-based refinement, a number of rules are selected as preliminary refinement for **Candidate Filtering**. These rules are easy implemented and fast in execution. Then, in informatics-based elimination, PMI (Point-wise Mutual Information) and entropy are calculated to select candidates of high co-occurrence rate and informative flexibility. Using informatics-based methods can greatly help narrow down the scope of MAW candidates and remove false positive cases. Lastly, search engine based validation is adopted to filter out low-vitality terms based on user links. This intellectual agent provide use cases about a candidate word as extra evidence. Details of these steps are described in the following subsections.

### 3.1 Rule-based Refinement

Brute-force based **Candidate Extraction** can ensure highest recall. Yet, it can create a substantial list of false positive candidates, such as the sub-component of a positive case: “啦A梦”, whose correct MAW should be “哆啦A梦”, *Doraemon*; and the under segmented token: “A股/反弹”, *rally of Shanghai SE Composite Index*, although the correct MAW should be “A股”, *Shanghai SE Composite Index*, etc. Below is a typical example of a user post in this dataset which includes a number of web-specific linguistic usages.

```

E2: #BMW赛车纪录片#
#亚洲公路摩托锦标赛珠海站全记录#
@UNIQ-王一博http://t.cn/EPdahkI
(#BMW Racing Documentary#Records Zhuhai
(in Asian Highway Motorcycle Championship.
@AX12FZ32 http://t.cn/EPdahkI)

```

As shown in E2, among all alphabetical chunks, many candidates are URL links, tags related to topics (surrounded by #), or user names (introduced by the “@” symbol). These alphabetical sequences is noise for MAWS and should be readily excluded from the final data using some simple rules. other false MAW candidates also demonstrate obvious patterns. For example, candidates of emoji (e.g. “QAQ”, “LOL”, “:P”, “T\_T”) are transformed symbols that encode no lexical meanings and shall be eliminated from the MAW list.

Using a set of 9 different pattern-based rules to filter out these unambiguous noises can largely reduce noisy data without compromising the coverage of the MAW lexicon. Detailed description of these patterns shall be introduced in Section 4.1.

### 3.2 Informatics-based Elimination

As will be shown in the evaluation that even after Rule-based Refinement, the candidate list it is still too large to be correct even by common sense. Informatics-based elimination works on this set of candidates to further remove noise.

Term-frequency (TF) is a commonly used metric to filter out low-occurrence candidates. However, using TF alone is insufficient to identify MAWs. For instance, both “A股”, *Shanghai SE Composite Index* and “A股/反弹”, *rally of Shanghai SE Composite Index* have high TF but only “A股” is a valid MAW. In this work, informatics-based methods are used to automatically filter the negative cases, including PMI for measuring the internal cohesion,

and entropy for measuring the external uncertainty of the candidates.

Point-wise mutual information (PMI) is proposed by Bouma (2009) to measure the co-occurrence probability of two variables. It is used to measure the internal “fixedness” of a word. Let  $w$  be an MAW candidate that consists of two components  $c_1, c_2$ . The PMI of  $w$  with respect to  $c_1$  and  $c_2$  can be calculated via Formula 1 given below.

$$PMI(c_1; c_2) = -\log\left(\frac{p(c_1, c_2)}{p(c_1) * p(c_2)}\right) \quad (1)$$

In practice, at least one component, denoted as  $c_a$  must contain alphabet character(s). If  $w$  consists of more than three components, we use the combination coordinated by  $c_a$ . For example, “哆啦/A/梦” *Doraemon* can be computed by using “哆啦A/梦” and “哆啦/A梦”. Formula 1 can be extended to Formula 2 to handle three components.

$$PMI(w) = \min(PMI(c_1; c_a), PMI(c_a; c_2)) \quad (2)$$

The threshold of PMI is experimentally set. Another dimension for identifying word boundaries is to use information entropy of its collocation environment. As proposed by He and Jun-Fang (2006), information entropy can be used to measure the uncertainty (flexibility) of a candidate’s environment, the larger the more flexible, and the more likely the candidate being a word. Consider the negative case of “素C” which only occurs in the context of “维生素C”, *Vitamin C* (entropy in this case is low). In contrast, the positive case “维生素C” occur in many different contexts: “补充/维生素C”, *Take Vitamin C*, “高剂量/维生素C”, *High-dosage Vitamin C*, “维生素C/对/感冒/有效”, *Vitamin C copes with colds*, etc. (entropy in this case is high). Let  $c_h$  and  $c_t$  be the respective head and tail components surrounding  $w$ . The head entropy of  $w$ , denoted by  $H(h)$ , is defined by Formula 3. The tail entropy  $H(t)$  can be obtained similarly. Based on Formula 3, the final entropy of  $w$  is obtained by  $\min(H(h), H(t))$ .

$$H(h) = - \sum p(c_h)_i * \log(p(c_h)_i) \quad (3)$$

### 3.3 Search Engine Validation

Search Engine Validation aims to further filter out candidate MAWs which are either less frequently used or in proper word forms that are not necessarily meaningful as lexical terms. A search engine such as Google, Bing and Baidu provide access to

a large knowledge base to validate the semantic information of a MAW candidate. Active MAW candidates with more links are more likely to carry proper semantic meanings. semantic information can help to exclude non-lexicon candidates. For instance, “UNIQ-王一博”, refers to *Wang Yi Bo*, a famous Chinese actor in the band “UNIQ”. The features of this false candidate can pass previous filtering methods perfectly. This indicates the need for a more intelligent validation scheme. As the data source in this work is Sina Weibo, it is more appropriate to use Baidu, the most popular search engine in China, as the knowledge agent for retrieving the validation evidence of the remaining candidates. Figure 2 is the flowchart of Search Engine Validation module.

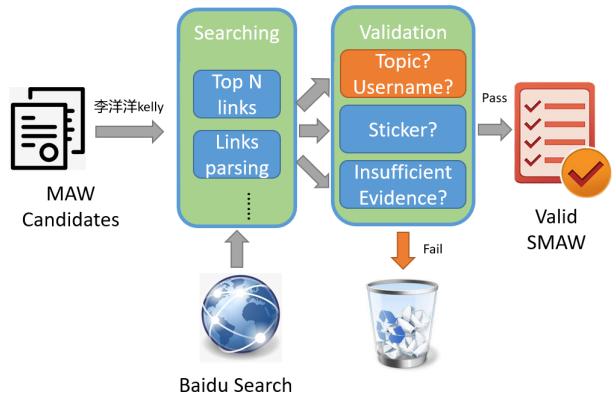


Figure 2: Flowchart of Search Engine Validation

Let us examine a user name as an example. “李洋洋kelly”, “Yangyang Li, Kelly” is a username combined with a Chinese name and an English nickname). The top  $N$  links are first collected as external evidence. The linked text is then cleaned and parsed to check whether this MAW candidate is meaningful. In the case of “李洋洋kelly” occurs only as “@李洋洋kelly”. Thus, it is validated as a username, not a real MAW. In addition to username checking, stickers and in sufficient occurrences are also used as indication of invalid MAWs.

## 4 Results and Evaluation

In our system, every filtering method is executed sequentially. Due to length limitation of this paper, we are giving the final selected parameters of our modules without showing the tuning process. The N-gram token window size of brute-force method in **Candidate Extraction** is set to 5 because most new terms are not longer than 5 as a common practice. In **Candidate Filtering**,

*LEN\_THRES* and *FREQ\_THRES* (detailed in Table 2) in rule-based refinement are tuned to 15 and 3, respectively. The upper bound of PMI and entropy in informatics-based elimination are experimentally set to -16.2 and 0.2, respectively. In search engine validation, we use the top 10 links as external evidence. If the number of valid links is less than 5, the corresponding MAW candidate is filtered out.

#### 4.1 Evaluation of SMAW

This section gives an estimate on the quality of SMAW in terms of **Accuracy**, **Candidate Size** and **Inter-rater agreement** through evaluation by human raters. As MAWs demonstrate a dynamic role in the Chinese lexicon, it is infeasible to refer to a full reference set for calculating Recall and Precision. That is the reason accuracy is used to measure quality of SMAW.

In the evaluation, three groups of SMAWs (100 each group, 300 in total) are randomly sampled from each step for the participants to judge the acceptance of the candidates. Raters are asked to make judgements and give 1 if they think a candidate is a MAW, or 0 otherwise. Then, Accuracy (Acc.) is calculated as the average of the three groups' acceptance rates. Incrementally, the Candidate Size (Size.) is also studied for each filtering method.

Inter-rater agreement among the three raters is also measured using Cohen's Kappa Coefficient (*K.*) (Kraemer, 2014). The evaluation results are given in Table 1.

Step	Method	Acc.	K.	Size.
1	BF	NA	.56	25,594k
2	+Rule-based	.22	.58	1,470k
3	+PMI	.62	.65	592k
4	+Entropy	.77	.70	32k
5	+Baidu	.82	<b>.78</b>	16k
B0	TF+Max.	.15	.59	1,935k

Table 1: The Evaluation Results

Starting from Brute-force, referred as BF, Table 1 summarizes the accumulative performances of using various metrics for candidate selection after each step. B0 is a baseline method that simply employs term frequency and the maximal sequence principle. For example, the maximal sequence principle will select “哆啦A梦”, *Doraemon* over components “啦A梦” or “A梦”. However, B0 is

more error-prone, For example, in “安全/使用/免费/WiFi”, *Safely use free wifi* where “免费WiFi”, *free wifi* shall be a positive instance.

In general, the accuracy increases when more filtering methods applied. It is worth mentioning that the accuracy shows a great boosting after using PMI and entropy, indicating the usefulness of informatics-based metrics for word identification. In addition, the incremental *K.* of each phase suggests the increased agreement methods the three raters by adopting the several metrics, especially after the Baidu search engine validation.

Compared with baseline method, our system makes use of a more reliable extraction approach that is obviously more effective for the identification of alphabetical words (Acc. = 0.82, *K.* = 0.78). The high accuracy score and agreement in the evaluation has proven the effectiveness of the extraction method, as well as demonstrating a good quality of the lexicon.

As for the candidate size, it can be observed that the candidate size drastically decreases after filtering methods. The total number of tokens obtained after brute-force candidate extraction reaches 25,594K, obviously too large and too noisy for direct use. After Rule-based Refinement, a set of 1,470k potential MAW candidates is obtained, only 5.7% of complete candidate collection. To provide more detail of rule-based refinement, Table 2 shows the process of constructing SMAW list of patterns used and the information on the reduction in data sizes.

By using PMI and entropy, 878k and 560k invalid MAW candidates are eliminated, respectively. The 97.8% reduction further narrow down the candidate set, only 33k candidates remain in the list. After processing this list based on search engine validation, the final collection of SMAWS has 16,207 tokens.

#### 4.2 The Lexical Characteristics

This section analyses the lexical properties of the SMAW lexicon. Comparisons between the SMAW list (“Web” hereinafter) and the MAWs in Huang and Liu (2017) (“Giga” hereinafter) will be made in terms of key vocabulary, length distribution, word formation types and lexical diversity so as to highlight the lexical differences of MAWs between social media and newspaper as well as the lexical development of alphabetical words in the recent two decades.

Rule	Description	Quantity
NONE	brute force candidates collection	25,594k
Topic	remove candidates with '#'	165k
Username	remove candidates with '@'	297k
No Chinese	remove candidates without Chinese character	1,302k
Too Short Length	remove candidates less than LEN_THRES characters	595k
Rare Occurrence	remove candidates which count less than FREQ_THRES	18,443k
English Expression	remove candidates contain two or more English words	1,421k
Symbol	remove candidates contain symbols such as '&' and '*'	419k
Emoji	remove candidates contain emoji such as "XDD"	193k
POS tag	remove candidates with invalid POS tag such as 'DET'	1k
ALL RULES	Remains after using all rule-based refinement	1,470k

Table 2: Noise Reduction Statistics by Rule-based Refinement.

### 4.2.1 Vocabulary

Figure 3 visualizes the top 50 MAW vocabularies of the two lexicons. The sizes of the words reflect its usage frequency.

It can be observed that the most frequent MAW in the Giga list is “B型” (B-type), while in the Web list, the most frequent MAW is “HOLD住” (To endure), which is a typical Internet neology. Moreover, most MAWs in Giga are disyllabic, e.g. “A型” (A-type) and “A级” (A-level), while SMAWs tend to be more lengthy, containing words of a wider range of syllables (e.g. “NBA全明星” (NBA all-star)). Specifically, MAWs in Giga show a dominant (rigid) pattern of “X类/型” (Type-X). However, in Web, MAWs have more Part-of-Speech diversity, including verbs (e.g. “Hold住”), nouns (e.g. “BB霜” (BB cream)), or adjectives (e.g. “牛X” (incredibly awesome)), indicating the trend of MAWs accounting for different grammatical roles in the Chinese language. Lastly, the lexical senses of Giga MAWs are more concentrated to the “type/classification” meaning, while MAWs in Web encode a wider range of meanings, including name entities, swear words, economics, entertainment, etc.

The above keyword differences reflect a dramatic change of MAWs at syllabic, lexical, grammatical and semantic levels in recent decades.

### 4.2.2 Length Distribution

The box-plots in Figure 4 give an overview of the length distribution of MAWs in Giga (Huang and Liu, 2017) and Web (SMAW).

As shown in Figure 4, MAWs in Web are much longer and more scattered than that in Giga. The mean length of MAWs in Giga is 2-3. But, the



Figure 3: Word clouds of MAWs in Web and Giga



Figure 4: Length distribution of MAWs in Giga and Web

mean length in SNAW is around 5. Overall, the MAWs in Web are distributed across a wider span. This may imply a tendency of code-mixing words being longer and richer in Modern Chinese.

#### 4.2.3 Word Formation

In line with the work of Huang and Liu (2017), word formation of MAWs is classified into four major types according to the positions of the A (alphabet) and C (character), including AC (e.g. “x-光”), CA (e.g. “牛b”), CAC (e.g. “程I青” (a Chinese Name)) and other types. The number of the four types of MAWs in Giga and Web is shown in Table 3 for comparison.

	AC	CA	CAC	Other	Total
Giga (pct)	665 <b>57.8%</b>	283 24.6%	185 16.1%	18 1.5%	1151 100.0%
Web (pct)	6971 43.0%	6994 <b>43.2%</b>	2242 13.8%	0 0.0%	16207 100.0%

Table 3: Word formation comparison

As highlighted in Table 3, the dominant type in Giga is AC, while CA is more prevalent in Web. Huang and Liu (2017) argued that the dominance of AC type with the modifier-modified compound structure in Chinese is because heads of nouns are usually right positioned (Sun, 2006). However, MAWs in Web have wider grammatical roles and more verbs are found in SMAW. Contrary to nouns, verbs are left headed, such as in “打call” (cheer up), where “打” (beat) is the head. In addition, cases like “维c” (Vitamin C), “双c” (double cores), and “最In” (Most popular) are headed on alphabets

instead of the Chinese character, indicating that heads are not necessarily positioned at the Chinese characters.

#### 4.2.4 Lexical Diversity

TTR (type–token ratio) is used to measure the lexical diversity/richness of a language (Durán et al., 2004). This metric is adopted here with normalized data (STTR), for measuring the lexical diversity of the MAWs in Giga and Web, as shown in Table 4.

Data	STTR	AC.STTR	CA.STTR
Web	14.53	16.9	12.3
Giga	8.77	7.6	15.2

Table 4: Lexical Diversity Comparison

Table 4 seems to suggest a reverse relation between the frequency of the MAW types and their lexical richness: the “AC” type is dominant in Giga, but it demonstrates a lower STTR; similarly, the “CA” type is dominant in Web, and it also shows a lower STTR. Overall, the Web MAWs show a richer vocabulary compared to the newspaper MAWs (Giga), indicating the higher productivity of social media language.

#### 4.3 The Corpus

In addition to the SMAW lexicon, we have also retrieved more than 200,000 sentences (around 2,000,000 tokens) for the 16,207 SMAW (each SMAW contains 10 or so sentences) to construct a SMAW corpus which can support code-mixing words inquiries.

一定(D) 要(D)	HOLD住(VA)	!
疯狂(D) 店庆(VA) 11天(Nd), 还(D) 能(D)	HOLD住(VA)	吗(T)
KITTY控(Na) 们(Na) 还(D)	HOLD住(VA)	吗(T)
微时代(Na), 大(A) 趋势(Na), 可得(VH)	HOLD住(VA)	!
亲(I)! 你(Nh) 要(D)	HOLD住(VA)	哦(T)
大家(Nh)	HOLD住(VA)	哦(T)
各位(Nes) 看官(Na) 要(D)	HOLD住(VA)	了(Di)

Interface 1: Corpus samples of “HOLD住” (KWIC)

The characters in the sentences are all transferred into simplified Chinese for consistency. All sentences are automatically segmented using Stanford CoreNLP<sup>1</sup> (Manning et al., 2014). The automatic word segmentation is enabled as the alphabetical words are pre-identified in our SMAW lexicon. With confirmed boundaries of the alphabetical

<sup>1</sup>[https://stanfordnlp.github.io/  
CoreNLP/](https://stanfordnlp.github.io/CoreNLP/)

words, it becomes an ordinary task of segmenting the remaining Chinese characters. On the basis of the raw sentences, we are building a concordance engine for loading the content of the corpus following the Chinese Word Sketch schema (Hong and Huang, 2006), which can support users' inquiries of word and grammatical collocations of code-mixing words. Samples of the corpus are shown in Interface 1.

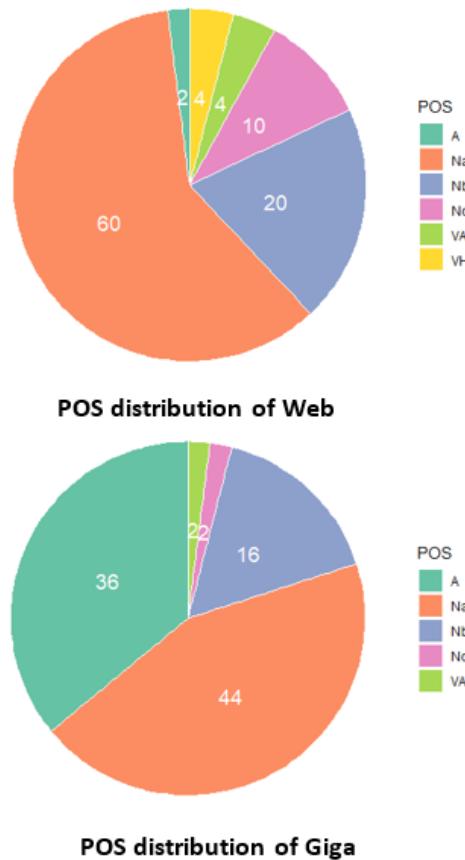


Figure 5: POS distribution of MAWs in Giga and Web

Besides, the corpus is undergoing a POS tagging process using the Academia Sinica segmentation and tagging system (Chen et al., 1996; Zhao et al., 2006) in order to support grammatical inquiries of linguistic accounts. Tagging is conducted automatically with manual post-checking on the SMAWs. The precision accuracy is estimated to be over 85%. Since tagging is still in progress, we provide the POS distribution<sup>2</sup> of the most frequent 50 SMAWs to show a general view of the grammatical distribution of popular SMAWs. Figure 5 shows the POS distribution of MAWs in Web and Giga for comparison purpose.

The POS distribution in Figure 5 shows that MAWs have developed a more salient role in the Chinese lexicon: from mainly nouns (Na, Nb, Nd) to verbs (VA, VH), from modifiers (A) to core lexical components (heads and arguments), and the graph demonstrates a more diversified lexical categories (more divisions and colorful) of new MAWs.

## 5 Conclusion and Future Work

This work uses social media platform (Sina Weibo) and search engine (Baidu) for collection and validation of code-mixing words to tackle the under-representation and identification problems of MAWs. The evaluation of the new Sina MAW dataset (SMAW), proves the high performance (Acc. = 0.82, K. = 0.78) of the proposed extraction method as well as the effectiveness our proposed candidate filtering techniques in terms of reducing number of noisy candidates. The contribution of this work is two-fold: it proposes an innovative method of leveraging the Web for MAW extraction without involvement of manual mediation, yet achieving promising performance in identifying out-of-vocabulary code-mixing words; it provides a unique MAW dataset and corresponding corpus which are most updated, scaled, structured and comprehensive for supporting linguistic inquiries of code-mixing words, as well as for facilitating related NLP tasks. The preliminary analysis to the lexical and grammatical characteristics of SMAWs and the corpus imply the development of code-mixing words into being a more important and diversified component in the Chinese lexicon. Future work will continue the annotation of the lexicon and the corpus with information of domains, sources, active time, semantic classes, etc., and conduct deeper linguistic analyses for uncovering the phonological and morpho-lexical characteristics of code-mixing words.

## Acknowledgments

We acknowledge the research grants from Hong Kong Polytechnic University (PolyU RTVU) and GRF grant (CERG PolyU 15211/14E, PolyU 152006/16E and PolyU 156086/18H). This work is also funded by the Post-doctoral project (no. 4-ZZKE) at the Hong Kong Polytechnic University.

<sup>2</sup><https://catalog.ldc.upenn.edu/LDC2009T14>

## References

- Gerlof Bouma. 2009. Normalized (pointwise) mutual information in collocation extraction. *Proceedings of GSCL*, pages 31–40.
- Jing-Shin Chang and Keh-Yih Su. 1997. An unsupervised iterative method for chinese new lexicon extraction. In *International Journal of Computational Linguistics & Chinese Language Processing, Volume 2, Number 2, August 1997*, pages 97–148.
- Keh-Jiann Chen, Chu-Ren Huang, Li-Ping Chang, and Hui-Li Hsu. 1996. Sinica corpus: Design methodology for balanced corpora. In *Proceedings of the 11th Pacific Asia Conference on Language, Information and Computation*, pages 167–176.
- Keh-Jiann Chen and Shing-Huan Liu. 1992. Word identification for mandarin chinese sentences. In *Proceedings of the 14th conference on Computational linguistics-Volume 1*, pages 101–107. Association for Computational Linguistics.
- Keh-Jiann Chen and Wei-Yun Ma. 2002. Unknown word extraction for chinese documents. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics.
- Gaël Dias. 2003. Multiword unit hybrid extraction. In *Proceedings of the ACL 2003 workshop on Multiword expressions: analysis, acquisition and treatment-Volume 18*, pages 41–48. Association for Computational Linguistics.
- Hongwei Ding, Yuanyuan Zhang, Hongchao Liu, and Chu-Ren Huang. 2017. A preliminary phonetic investigation of alphabetic words in mandarin chinese. In *INTERSPEECH*, pages 3028–3032.
- Pilar Durán, David Malvern, Brian Richards, and Ngoni Chipere. 2004. Developmental trends in lexical diversity. *Applied Linguistics*, 25(2):220–242.
- Jia-Fei Hong and Chu-Ren Huang. 2006. Using chinese gigaword corpus and chinese word sketch in linguistic research. In *Proceedings of the 20th Pacific Asia Conference on Language, Information and Computation*, pages 183–190.
- Chu-Ren Huang. 2009. Tagged chinese gigaword version 2.0, ldc2009t14. *Linguistic Data Consortium*.
- Chu-Ren Huang and Hongchao Liu. 2017. Corpus-based automatic extraction and analysis of mandarin alphabetic words (in chinese). *Journal of Yunnan Teachers University. Philosophy and social science section*.
- Chu-Ren Huang, Petr Šimon, Shu-Kai Hsieh, and Laurent Prévot. 2007. Rethinking chinese word segmentation: tokenization, character classification, or wordbreak identification. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 69–72.
- Shaohua Jiang and Yanzhong Dang. 2007. Automatic extraction of new-domain terms containing chinese lettered words (in chinese). *Computing Engineering*, 33(2):47–49.
- Ksenia Kozha. 2012. Chinese via english: A case study of “lettered-words” as a way of integration into global communication. In *Chinese Under Globalization: Emerging Trends in Language Use in China*, pages 105–125. World Scientific.
- Helena C Kraemer. 2014. Kappa coefficient. *Wiley StatsRef: Statistics Reference Online*, pages 1–4.
- Yongquan Liu. 1994. Survey on chinese lettered words (in chinese). *Language Planning*, (10):7–9.
- Yongquan Liu. 2002. The issue of lettered words in chinese. *Applied Linguistics*, 1:8S–90.
- Ka Yee Lun. 2013. Morphological structure of the chinese lettered words. *University of Washington Working Papers in Linguistics*.
- Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60.
- Ruiqin Miao. 2005. *Loanword adaptation in Mandarin Chinese: Perceptual, phonological and sociolinguistic factors*. Ph.D. thesis, Stony Brook University.
- Dong Nguyen and Leonie Cornips. 2016. Automatic detection of intra-word code-switching. In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 82–86.
- He Ren and Jun-fang Zeng. 2006. A chinese word extraction algorithm based on information entropy. *Journal of Chinese Information Processing*, 20(5):40–43.
- Helena Riha. 2010. Lettered words in chinese: Roman letters as morpheme-syllables.
- Helena Riha and Kirk Baker. 2010. Using roman letters to create words in chinese. In *Variation and Change in Morphology: Selected papers from the 13th International Morphology Meeting, Vienna, February 2008*, volume 310, page 193. John Benjamins Publishing.
- Xinchun Su and Xiaofang Wu. 2013. Vitality and limitation of chinese lettered words (in chinese). *Journal of Beihua University(Social Sciences)*, 2.
- Chaofen Sun. 2006. *Chinese: A linguistic introduction*. Cambridge University Press.
- Li Hai Tan, Angela R Laird, Karl Li, and Peter T Fox. 2005. Neuroanatomical correlates of phonological processing of chinese characters and alphabetic words: A meta-analysis. *Human brain mapping*, 25(1):83–91.

Nianwen Xue and Libin Shen. 2003. Chinese word segmentation as lmr tagging. In *Proceedings of the second SIGHAN workshop on Chinese language processing-Volume 17*, pages 176–179. Association for Computational Linguistics.

Xiaocong Xue. 2007. A review on studies of lettered-words in contemporary chinese. *Chinese Language Learning*, 2.

Haijun Zhang, Heyan Huang, Chaoyong Zhu, and Shumin Shi. 2010. A pragmatic model for new chinese word extraction. In *Proceedings of the 6th International Conference on Natural Language Processing and Knowledge Engineering (NLPKE-2010)*, pages 1–8. IEEE.

Tiewen Zhang. 2005. Study of the word family ‘x-ray’ in chinese (in chinese). *Terminology Standardization & Information Technology*, 1.

Tiewen Zhang. 2013. The use of chinese lettered-words is a normal phenomenon of language contact. (in chinese). *Journal of Beihua University(Social Sciences)*, 2.

Hai Zhao, Changning Huang, and Mu Li. 2006. An improved chinese word segmentation system with conditional random field. In *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, pages 162–165.

Zezi Zheng, Pu Zhang, and Jianguo Yang. 2005. Corpus-based extraction of chinese lettered words (in chinese). *Journal of Chinese Information Processing*, 19(2):79–86.

# IndoNLU: Benchmark and Resources for Evaluating Indonesian Natural Language Understanding

Bryan Wilie<sup>1\*</sup>, Karissa Vincentio<sup>2\*</sup>, Genta Indra Winata<sup>3\*</sup>, Samuel Cahyawijaya<sup>3\*</sup>, Xiaohong Li<sup>4</sup>, Zhi Yuan Lim<sup>4</sup>, Sidik Soleman<sup>5</sup>, Rahmad Mahendra<sup>6</sup>, Pascale Fung<sup>3</sup>, Syafri Bahar<sup>4</sup>, Ayu Purwarianti<sup>1,5</sup>

<sup>1</sup>Institut Teknologi Bandung    <sup>2</sup>Universitas Multimedia Nusantara

<sup>3</sup>The Hong Kong University of Science and Technology

<sup>4</sup>Gojek    <sup>5</sup>Prosa.ai    <sup>6</sup>Universitas Indonesia

{bryanwifie92, karissavin}@gmail.com, {giwinata, scahyawijaya}@connect.ust.hk

## Abstract

Although Indonesian is known to be the fourth most frequently used language over the internet, the research progress on this language in natural language processing (NLP) is slow-moving due to a lack of available resources. In response, we introduce the first-ever vast resource for training, evaluation, and benchmarking on Indonesian natural language understanding (IndoNLU) tasks. IndoNLU includes twelve tasks, ranging from single sentence classification to pair-sentences sequence labeling with different levels of complexity. The datasets for the tasks lie in different domains and styles to ensure task diversity. We also provide a set of Indonesian pre-trained models (IndoBERT) trained from a large and clean Indonesian dataset (Indo4B) collected from publicly available sources such as social media texts, blogs, news, and websites. We release baseline models for all twelve tasks, as well as the framework for benchmark evaluation, thus enabling everyone to benchmark their system performances.

## 1 Introduction

Following the notable success of contextual pre-trained language methods (Peters et al., 2018; Devlin et al., 2019), several benchmarks to gauge the progress of general-purpose NLP research, such as GLUE (Wang et al., 2018), SuperGLUE (Wang et al., 2019), and CLUE (Xu et al., 2020), have been proposed. These benchmarks cover a large range of tasks to measure how well pre-trained models achieve compared to humans. However, these metrics are limited to high-resource languages, such as English and Chinese, that already have existing datasets available and are accessible to the research community. Most languages, by contrast, suffer from limited data collection and low awareness of

published data for research. One of the languages which suffer from this resource scarcity problem is Indonesian.

Indonesian is the fourth largest language used over the internet, with around 171 million users across the globe.<sup>1</sup> Despite a large amount of Indonesian data available over the internet, the advancement of NLP research in Indonesian is slow-moving. This problem occurs because available datasets are scattered, with a lack of documentation and minimal community engagement. Moreover, many existing studies in Indonesian NLP do not provide codes and test splits, making it impossible to reproduce results.

To address the data scarcity problem, we propose the first-ever Indonesian natural language understanding benchmark, IndoNLU, a collection of twelve diverse tasks. The tasks are mainly categorized based on the input, such as single-sentences and sentence-pairs, and objectives, such as sentence classification tasks and sequence labeling tasks. The benchmark is designed to cater to a range of styles in both formal and colloquial Indonesian, which are highly diverse. We collect a range of datasets from existing works: an emotion classification dataset (Saputri et al., 2018), QA factoid dataset (Purwarianti et al., 2007), sentiment analysis dataset (Purwarianti and Crisdayanti, 2019), aspect-based sentiment analysis dataset (Ilmania et al., 2018; Azhar et al., 2019), part-of-speech (POS) tag dataset (Dinakaramani et al., 2014; Hoesen and Purwarianti, 2018), named entity recognition (NER) dataset (Hoesen and Purwarianti, 2018), span extraction dataset (Mahfuzh et al., 2019; Septiandri and Sutiono, 2019; Fernando et al., 2019), and textual entailment dataset (Setya and Mahendra, 2018). It is difficult to compare model performance since there is no official

\*These authors contributed equally.

<sup>1</sup><https://www.internetworldstats.com/stats3.htm>

split of information for existing datasets. Therefore we standardize the benchmark by resplitting the datasets on each task for reproducibility purposes. To expedite the modeling and evaluation processes for this benchmark, we present samples of the model pre-training code and a framework to evaluate models in all downstream tasks. We will publish the score of our benchmark on a publicly accessible leaderboard to provide better community engagement and benchmark transparency.

To further advance Indonesian NLP research, we collect around four billion words from Indonesian preprocessed text data ( $\approx 23$  GB), as a new standard dataset, called `Indo4B`, for self-supervised learning. The dataset comes from sources like online news, social media, Wikipedia, online articles, subtitles from video recordings, and parallel datasets. We then introduce an Indonesian BERT-based model, `IndoBERT`, which is trained on our `Indo4B` dataset. We also introduce another `IndoBERT` variant based on the ALBERT model (Lan et al., 2020), called `IndoBERT-lite`. The two variants of `IndoBERT` are used as baseline models in the `IndoNLU` benchmark. In this work, we also extensively compare our `IndoBERT` models to different pre-trained word embeddings and existing multilingual pre-trained models, such as Multilingual BERT (Devlin et al., 2019) and XLM-R (Conneau et al., 2019), to measure their effectiveness. Results show that our pre-trained models outperform most of the existing pre-trained models.

## 2 Related Work

**Benchmarks** GLUE (Wang et al., 2018) is a multi-task benchmark for natural language understanding (NLU) in the English language. It consists of nine tasks: single-sentence input, semantic similarity detection, and natural language inference (NLI) tasks. GLUE’s harder counterpart SuperGLUE (Wang et al., 2019) covers question answering, NLI, co-reference resolution, and word sense disambiguation tasks. CLUE (Xu et al., 2020) is a Chinese NLU benchmark that includes a test set designed to probe a unique and specific linguistic phenomenon in the Chinese language. It consists of eight diverse tasks, including single-sentence, sentence-pair, and machine reading comprehension tasks. FLUE (Le et al., 2019) is an evaluation NLP benchmark for the French language which is divided into six different task categories: text classification, paraphrasing, NLI, parsing, POS tagging,

and word sense disambiguation.

**Contextual Language Models** In recent years, contextual pre-trained language models have shown a major breakthrough in NLP, starting from ELMo (Peters et al., 2018). With the emergence of the transformer model (Vaswani et al., 2017), Devlin et al. (2019) proposed BERT, a faster architecture to train a language model that eliminates recurrences by applying a multi-head attention layer. Liu et al. (2019) later proposed RoBERTa, which improves the performance of BERT by applying dynamic masking, increasing the batch size, and removing the next-sentence prediction. Lan et al. (2020) proposed ALBERT, which extends the BERT model by applying factorization and weight sharing to reduce the number of parameters and time.

Many research studies have introduced contextual pre-trained language models on languages other than English. Cui et al. (2019) introduced the Chinese BERT and RoBERTa models, while Martin et al. (2019) and Le et al. (2019) introduced CamemBERT and FLAUBert respectively, which are BERT-based models for the French language. Devlin et al. (2019) introduced the Multilingual BERT model, a BERT model trained on monolingual Wikipedia data in many languages. Meanwhile, Lample and Conneau (2019) introduced XLM, a cross-lingual pre-trained language model that uses parallel data as a new translation masked loss to improve the cross-linguality. Finally, Conneau et al. (2019) introduced XLM-R, a RoBERTa-based XLM model.

## 3 IndoNLU Benchmark

In this section, we describe our benchmark as four components. Firstly, we introduce the 12 tasks in `IndoNLU` for Indonesian natural language understanding. Secondly, we introduce a large-scale Indonesian dataset for self-supervised pre-training models. Thirdly, we explain the various kinds of baseline models used in our `IndoNLU` benchmark. Lastly, we describe the evaluation metric used to standardize the scoring over different models in our `IndoNLU` benchmark.

### 3.1 Downstream Tasks

The `IndoNLU` downstream tasks covers 12 tasks divided into four categories: (a) single-sentence classification, (b) single-sentence sequence-tagging, (c) sentence-pair classification, and (d)

Dataset	Train	Valid	Test	Task Description	#Label	#Class	Domain	Style
Single-Sentence Classification Tasks								
EmoT <sup>†</sup>	3,521	440	442	emotion classification	1	5	tweets	colloquial
SmSA	11,000	1,260	500	sentiment analysis	1	3	general	colloquial
CASA	810	90	180	aspect-based sentiment analysis	6	3	automobile	colloquial
HoASA <sup>†</sup>	2,283	285	286	aspect-based sentiment analysis	10	4	hotel	colloquial
Sentence-Pair Classification Tasks								
WReTE <sup>†</sup>	300	50	100	textual entailment	1	2	wiki	formal
Single-Sentence Sequence Labeling Tasks								
POSP <sup>†</sup>	6,720	840	840	part-of-speech tagging	1	26	news	formal
BaPOS	8,000	1,000	1,029	part-of-speech tagging	1	41	news	formal
TermA	3,000	1,000	1,000	span extraction	1	5	hotel	colloquial
KEPS	800	200	247	span extraction	1	3	banking	colloquial
NERGrit <sup>†</sup>	1,672	209	209	named entity recognition	1	7	wiki	formal
NERP <sup>†</sup>	6,720	840	840	named entity recognition	1	11	news	formal
Sentence-Pair Sequence Labeling Tasks								
FacQA	2,495	311	311	span extraction	1	3	news	formal

Table 1: Task statistics and descriptions. <sup>†</sup>We create new splits for the dataset.

sentence-pair sequence labeling. The data samples for each task are shown in Appendix A.

### 3.1.1 Single-Sentence Classification Tasks

**EmoT** An emotion classification dataset collected from the social media platform Twitter (Saputri et al., 2018). The dataset consists of around 4000 Indonesian colloquial language tweets, covering five different emotion labels: anger, fear, happiness, love, and sadness.

**SmSA** This sentence-level sentiment analysis dataset (Purwarianti and Crisdayanti, 2019) is a collection of comments and reviews in Indonesian obtained from multiple online platforms. The text was crawled and then annotated by several Indonesian linguists to construct this dataset. There are three possible sentiments on the SmSA dataset: positive, negative, and neutral.

**CASA** An aspect-based sentiment analysis dataset consisting of around a thousand car reviews collected from multiple Indonesian online automobile platforms (Ilmania et al., 2018). The dataset covers six aspects of car quality. We define the task to be a multi-label classification task, where each label represents a sentiment for a single aspect with three possible values: positive, negative, and neutral.

**HoASA** An aspect-based sentiment analysis dataset consisting of hotel reviews collected from the hotel aggregator platform, AiryRooms (Azhar

et al., 2019).<sup>2</sup> The dataset covers ten different aspects of hotel quality. Similar to the CASA dataset, each review is labeled with a single sentiment label for each aspect. There are four possible sentiment classes for each sentiment label: positive, negative, neutral, and positive-negative. The positive-negative label is given to a review that contains multiple sentiments of the same aspect but for different objects (e.g., cleanliness of bed and toilet).

### 3.1.2 Sentence-Pair Classification Task

**WReTE** The Wiki Revision Edits Textual Entailment dataset (Setya and Mahendra, 2018) consists of 450 sentence pairs constructed from Wikipedia revision history. The dataset contains pairs of sentences and binary semantic relations between the pairs. The data are labeled as entailed when the meaning of the second sentence can be derived from the first one, and not entailed otherwise.

### 3.1.3 Single-Sentence Sequence Labeling Tasks

**POSP** This Indonesian part-of-speech tagging (POS) dataset (Hoesen and Purwarianti, 2018) is collected from Indonesian news websites. The dataset consists of around 8000 sentences with 26 POS tags. The POS tag labels follow the Indonesian Association of Computational Linguistics (INACL) POS Tagging Convention.<sup>3</sup>

<sup>2</sup><https://github.com/annisanurulazhar/absa-playground>

<sup>3</sup><http://inacl.id/inacl/wp-content/uploads/2017/06/INACL-POS-Tagging-Convention-26-Mei.pdf>

Model	#Params	#Layers	#Heads	Emb. Size	Hidden Size	FFN Size	Language Type	Pre-train Emb. Type
Scratch	15.1M	6	10	300	300	3072	Mono	-
fastText-cc-id	15.1M	6	10	300	300	3072	Mono	Word Emb.
fastText-indo4b	15.1M	6	10	300	300	3072	Mono	Word Emb.
IndoBERT-lite <sub>BASE</sub>	11.7M	12	12	128	768	3072	Mono	Contextual
IndoBERT <sub>BASE</sub>	124.5M	12	12	768	768	3072	Mono	Contextual
IndoBERT-lite <sub>LARGE</sub>	17.7M	24	16	128	1024	4096	Mono	Contextual
IndoBERT <sub>LARGE</sub>	335.2M	24	16	1024	1024	4096	Mono	Contextual
mBERT	167.4M	12	12	768	768	3072	Multi	Contextual
XLM-R <sub>BASE</sub>	278.7M	12	12	768	768	3072	Multi	Contextual
XLM-R <sub>LARGE</sub>	561.0M	24	16	1024	1024	4096	Multi	Contextual
XLM-MLM <sub>LARGE</sub>	573.2M	16	16	1280	1280	5120	Multi	Contextual

Table 2: The details of baseline models used in IndoNLU benchmark

**BaPOS** This POS tagging dataset (Dinakaramani et al., 2014) contains about 1000 sentences, collected from the PAN Localization Project.<sup>4</sup> In this dataset, each word is tagged by one of 23 POS tag classes.<sup>5</sup> Data splitting used in this benchmark follows the experimental setting used by Kurniawan and Aji (2018).

**TermA** This span-extraction dataset is collected from the hotel aggregator platform, AiryRooms (Septiandri and Sutiono, 2019; Fernando et al., 2019).<sup>6</sup> The dataset consists of thousands of hotel reviews, which each contain a span label for aspect and sentiment words representing the opinion of the reviewer on the corresponding aspect. The labels use Inside-Outside-Beginning (IOB) tagging representation with two kinds of tags, aspect and sentiment.

**KEPS** This keyphrase extraction dataset (Mahfuzh et al., 2019) consists of text from Twitter discussing banking products and services and is written in the Indonesian language. A phrase containing important information is considered a keyphrase. Text may contain one or more keyphrases since important phrases can be located at different positions. The dataset follows the IOB chunking format, which represents the position of the keyphrase.

**NERGrit** This NER dataset is taken from the Grit-ID repository,<sup>7</sup> and the labels are spans in IOB chunking representation. The dataset consists of

three kinds of named entity tags, PERSON (name of person), PLACE (name of location), and ORGANIZATION (name of organization).

**NERP** This NER dataset (Hoesen and Purwarianti, 2018) contains texts collected from several Indonesian news websites. There are five labels available in this dataset, PER (name of person), LOC (name of location), IND (name of product or brand), EVT (name of the event), and FNB (name of food and beverage). Similar to the TermA dataset, the NERP dataset uses the IOB chunking format.

### 3.1.4 Sentence-Pair Sequence Labeling Task

**FacQA** The goal of the FacQA dataset is to find the answer to a question from a provided short passage from a news article (Purwarianti et al., 2007). Each row in the FacQA dataset consists of a question, a short passage, and a label phrase, which can be found inside the corresponding short passage. There are six categories of questions: date, location, name, organization, person, and quantitative.

## 3.2 Indo4B Dataset

Indonesian NLP development has struggled with the availability of data. To cope with this issue, we provide a large-scale dataset called Indo4B for building a self-supervised pre-trained model. Our self-supervised dataset consists of around 4B words, with around 250M sentences. The Indo4B dataset covers both formal and colloquial Indonesian sentences compiled from 12 datasets, of which two cover Indonesian colloquial language, eight cover formal Indonesian language, and the rest have a mixed style of both colloquial and formal. The statistics of our large-scale dataset can be

<sup>4</sup><http://www.panl10n.net/>

<sup>5</sup><http://bahasa.cs.ui.ac.id/postag/downloads/Tagset.pdf>

<sup>6</sup>[https://github.com/jordhy97/final\\_project](https://github.com/jordhy97/final_project)

<sup>7</sup><https://github.com/grit-id/nergrit-corpus>

Dataset	# Words	# Sentences	Size	Style	Source
OSCAR (Ortiz Suárez et al., 2019)	2,279,761,186	148,698,472	14.9 GB	mixed	OSCAR
CoNLLu Common Crawl (Ginter et al., 2017)	905,920,488	77,715,412	6.1 GB	mixed	LINDAT/CLARIAH-CZ
OpenSubtitles (Lison and Tiedemann, 2016)	105,061,204	25,255,662	664.8 MB	mixed	OPUS OpenSubtitles
Twitter Crawl <sup>2</sup>	115,205,737	11,605,310	597.5 MB	colloquial	Twitter
Wikipedia Dump <sup>1</sup>	76,263,857	4,768,444	528.1 MB	formal	Wikipedia
Wikipedia CoNLLu (Ginter et al., 2017)	62,373,352	4,461,162	423.2 MB	formal	LINDAT/CLARIAH-CZ
Twitter UI <sup>2</sup> (Saputri et al., 2018)	16,637,641	1,423,212	88 MB	colloquial	Twitter
OPUS JW300 (Agić and Vulić, 2019)	8,002,490	586,911	52 MB	formal	OPUS
Tempo <sup>3</sup>	5,899,252	391,591	40.8 MB	formal	ILSP
Kompas <sup>3</sup>	3,671,715	220,555	25.5 MB	formal	ILSP
TED	1,483,786	111,759	9.9 MB	mixed	TED
BPPT	500,032	25,943	3.5 MB	formal	BPPT
Parallel Corpus	510,396	35,174	3.4 MB	formal	PAN Localization
TALPCo (Nomoto et al., 2018)	8,795	1,392	56.1 KB	formal	Tokyo University
Frog Storytelling (Moeljadi, 2012)	1,545	177	10.1 KB	mixed	Tokyo University
<b>TOTAL</b>	3,581,301,476	275,301,176	23.43 GB		

Table 3: Indo4B dataset statistics. <sup>1</sup> <https://dumps.wikimedia.org/backup-index.html>. <sup>2</sup> We crawl tweets from Twitter. The Twitter data will not be shared publicly due to restrictions of the Twitter Developer Policy and Agreement. <sup>3</sup> <https://ilps.science.uva.nl/>.

found in Table 3. We share the datasets that are listed in the table, except for those from Twitter due to restrictions of the Twitter Developer Policy and Agreement. The details of Indo4B dataset sources are shown in Appendix B.

### 3.3 Baselines

In this section, we explain the baseline models and the fine-tuning settings that we use in the IndoNLU benchmark.

#### 3.3.1 Models

We provide a diverse set of baseline models, from a non-pre-trained model (scratch), to a word-embedding-based model, to contextualized language models. For the word-embeddings-based model, we use an existing fastText model trained on the Indonesian Common Crawl (CC-ID) dataset (Joulin et al., 2016; Grave et al., 2018).

**fastText** We build a fastText model with our large-scale self-supervised dataset, Indo4B, for comparison with the CC-ID fastText model and contextualized language model. For the models above and the fastText model, we use the transformer architecture (Vaswani et al., 2017). We experiment with different numbers of layers, 2, 4, and 6, for the transformer encoder. For the fastText model, we first pre-train the fastText embeddings with skipgram word representation and produce a 300-dimensional embedding vector. We then generate all required embeddings for each downstream task from the pre-trained fastText embeddings and

cover all words in the vocabulary.

**Contextualized Language Models** We build our own Indonesian BERT and ALBERT models, named IndoBERT and IndoBERT-lite, respectively, in both base and large sizes. The details of our IndoBERT and IndoBERT-lite models are explained in Section 4. Aside from a monolingual model, we also provide multilingual model baselines such as Multilingual BERT (Devlin et al., 2019), XLM (Lample and Conneau, 2019), and XLM-R (Conneau et al., 2019). The details of each model are shown in Table 2.

#### 3.3.2 Fine-tuning Settings

We fine-tune a pre-trained model for each task with initial learning with a range of learning rates [1e-5, 4e-5]. We apply a decay rate of [0.8, 0.9] for every epoch, and sample each batch with a size of 16 for all datasets except FacQA and POSP, for which we use a batch size of 8. To establish a benchmark, we keep a fixed setting, and we use an early stop on the validation score to choose the best model. The details of the fine-tuning hyperparameter settings used are shown in Appendix D.

### 3.4 Evaluation Metrics

We use the F1 score to measure the evaluation performance of all tasks. For the binary and multi-label classification tasks, we measure the macro-averaged F1 score by taking the top-1 prediction from the model. For the sequence labeling task, we calculate word-level sequence labeling macro-

Model	Maximum Sequence Length = 128				Maximum Sequence Length = 512			
	Batch Size	Learning Rate	Steps	Duration (Hr.)	Batch Size	Learning Rate	Steps	Duration (Hr.)
IndoBERT-lite <sub>BASE</sub>	4096	0.00176	112.5 K	38	1024	0.00088	50 K	23
IndoBERT <sub>BASE</sub>	256	0.00002	1 M	35	256	0.00002	68 K	9
IndoBERT-lite <sub>LARGE</sub>	1024	0.00044	500 K	134	256	0.00044	129 K	45
IndoBERT <sub>LARGE</sub>	256	0.0001	1 M	89	128	0.00008	120 K	32

Table 4: Hyperparameters and training duration for IndoBERT model pre-training.

averaged F1-score for all models by following the sequence labeling evaluation method described in the CoNLL evaluation script. We calculate two mean F1-scores separately for classification and sequence labeling tasks to evaluate models on our IndoNLU benchmark.

## 4 IndoBERT

In this section, we describe the details of our Indonesian contextualized models, IndoBERT and IndoBERT-lite, which are trained using our Indo4B dataset. We elucidate the extensive details of the models’ development, first the dataset preprocessing, followed by the pre-training setup.

### 4.1 Preprocessing

**Dataset Preparation** To get the most beneficial next sentence prediction task training from the Indo4B dataset, we do either a paragraph separation or line separation if we notice document separator absence in the dataset. This document separation is crucial as it is used in the BERT architecture to extract long contiguous sequences (Devlin et al., 2019). A separation between sentences with a new line is also required to differentiate each sentence. These are used by BERT to create input embeddings out of sentence pairs that are compacted into a single sequence. We specify the number of duplication factors for each of the datasets differently due to the various formats of the datasets that we collected. We create duplicates on datasets with the end of document separators with a higher duplication factor. The preprocessing method is applied in both the IndoBERT and IndoBERT-lite models.

We keep the original form of a word to hold its contextual information since Indonesian words are built with rich morphological operations, such as compounding, affixation, and reduplication (Pisceldo et al., 2008). In addition, this setting is also suitable for contextual pre-training models that leverage inflections to improve the sentence-level representations.(Kutuzov and Kuzmenko, 2019)

Twitter data contains specific details, such as usernames, hashtags, emails, and URL hyperlinks. To preserve privacy and also to reduce noise, this private information in the Twitter UI dataset (Saputri et al., 2018) is masked into generics tokens such as <username>, <hashtag>, <email> and <links>. On the other hand, this information is discarded in the larger Twitter Crawl dataset.

**Vocabulary** For both the IndoBERT and the IndoBERT-lite models, we utilize Sentence-Piece (Kudo and Richardson, 2018) with a byte pair encoding (BPE) tokenizer as the vocabulary generation method. We use a vocab size of 30.522 for the IndoBERT models and vocab size of 30.000 for the IndoBERT-lite models.

### 4.2 Pre-training Setup

All IndoBERT models are trained on TPUs in two phases. In the first phase, we train the models with a maximum sequence length of 128. The training takes around 35, 89, 38 and 134 hours on IndoBERT<sub>BASE</sub>, IndoBERT<sub>LARGE</sub>, IndoBERT-lite<sub>BASE</sub>, and IndoBERT-lite<sub>LARGE</sub>, respectively. In the second phase, we continue the training of the IndoBERT models with a maximum sequence length of 512. It takes 9, 32, 23 and 45 hours on IndoBERT<sub>BASE</sub>, IndoBERT<sub>LARGE</sub>, IndoBERT-lite<sub>BASE</sub>, and IndoBERT-lite<sub>LARGE</sub>, respectively. The details of the pre-training hyperparameter settings are shown in Appendix D.

**IndoBERT** We use a batch size of 256 and a learning rate of 2e-5 in both training phases for IndoBERT<sub>BASE</sub>, and we adjust the learning rate to 1e-4 for IndoBERT<sub>LARGE</sub> to stabilize the training. Due to memory limitation, we scale down the batch size to 128 and the learning rate to 8e-5 in the second phase of the training, with a number of training steps adapted accordingly. The base and large models are trained using the masked language modeling loss. We limit the maximum prediction per sequence into 20 tokens.

Model	Classification						Sequence Labeling							
	EmoT	SmSA	CASA	HoASA	WReTE	Avg	POSP	BaPOS	TermA	KEPS	NERGrit	NERP	FacQA	Avg
Scratch	57.31	67.35	67.15	76.28	64.35	66.49	86.78	70.24	70.36	39.40	5.80	30.66	5.00	44.03
fastText-cc-id	65.36	76.92	79.02	85.32	67.36	74.79	94.35	79.85	76.12	56.39	37.32	46.46	15.29	57.97
fastText-indo4b	69.23	82.13	82.20	85.88	60.42	75.97	94.94	81.77	74.43	56.70	38.69	46.79	14.65	58.28
mBERT	67.30	84.14	72.23	84.63	84.40	78.54	91.85	83.25	89.51	64.31	75.02	69.27	61.29	76.36
XLM-MLM	65.75	86.33	82.17	88.89	64.35	77.50	95.87	88.40	90.55	65.35	74.75	75.06	62.15	78.88
XLM-R <sub>BASE</sub>	71.15	91.39	91.71	91.57	79.95	85.15	95.16	84.64	90.99	68.82	79.09	75.03	64.58	79.76
XLM-R <sub>LARGE</sub>	78.51	92.35	92.40	94.27	83.82	88.27	92.73	87.03	91.45	70.88	78.26	78.52	74.61	81.92
IndoBERT-lite <sub>BASE</sub> <sup>†</sup>	73.88	90.85	89.68	88.07	82.17	84.93	91.40	75.10	89.29	69.02	66.62	46.58	54.99	70.43
+ phase two	72.27	90.29	87.63	87.62	83.62	84.29	90.05	77.59	89.19	69.13	66.71	50.52	49.18	70.34
IndoBERT <sub>BASE</sub> <sup>†</sup>	75.48	87.73	93.23	92.07	78.55	85.41	95.26	87.09	90.73	70.36	69.87	75.52	53.45	77.47
+ phase two	76.28	87.66	93.24	92.70	78.68	85.71	95.23	85.72	91.13	69.17	67.42	75.68	57.06	77.34
IndoBERT-lite <sub>LARGE</sub>	75.19	88.66	90.99	89.53	78.98	84.67	91.56	83.74	90.23	67.89	71.19	74.37	65.50	77.78
+ phase two	70.80	88.61	88.13	91.05	85.41	84.80	94.53	84.91	90.72	68.55	73.07	74.89	62.87	78.51
IndoBERT <sub>LARGE</sub>	77.08	92.72	95.69	93.75	82.91	88.43	95.71	90.35	91.87	71.18	77.60	79.25	62.48	81.21
+ phase two	79.47	92.03	94.94	93.38	80.30	88.02	95.34	87.36	92.14	71.27	76.63	77.99	68.09	81.26

Table 5: Results of baseline models with best performing configuration on the IndoNLU benchmark. Extensive experimental results are shown in Appendix E. Bold numbers are the best results among all. <sup>†</sup>The IndoBERT models are trained using two training phases.

**IndoBERT-lite** We follow the ALBERT pre-training hyperparameters setup (Lan et al., 2020) to pre-train the IndoBERT-lite models. We limit the maximum prediction per sequence into 20 tokens on the models, pre-training with whole word masked loss. We train the base model with a batch size of 4096 in the first phase, and 1024 in the second phase. Since we have a limitation in computation power, we use a smaller batch size of 1024 in the first phase and 256 in the second phase in training our large model.

## 5 Results and Analysis

In this section, we show the results of the IndoNLU benchmark and analyze the performance of our models in terms of downstream tasks score and performance-space trade-off. In addition, we show an analysis of the effectiveness of using our collected data compared to existing baselines.

### 5.1 Benchmark Results

**Overall Performance** As mentioned in Section 3, we fine-tune all baseline models mentioned in Section 3.3, and evaluate the model performance over all tasks, grouped into two categories, classification and sequence labeling. We can see in Table 5, that IndoBERT<sub>LARGE</sub>, XLM-R<sub>LARGE</sub>, and IndoBERT<sub>BASE</sub> achieve the top-3 best performance results on the classification tasks, and XLM-R<sub>LARGE</sub>, IndoBERT<sub>LARGE</sub>, and XLM-R<sub>BASE</sub> achieve the top-3 best performance results on the sequence labeling tasks. The experimental results also suggest that larger models have a performance advantage over smaller models. It is also evident

that all pre-trained models outperform the scratch model, which shows the effectiveness of model pre-training. Another interesting observation is that all contextualized pre-trained models outperform word embeddings-based models by significant margins. This shows the superiority of the contextualized embeddings approach over the word embeddings approach.

### 5.2 Performance-Space Trade-off

Figure 1 shows the model performance with respect to the number of parameters. We can see two large clusters. On the bottom left, the scratch and fastText models appear, and they have the lowest F1 scores and the least floating points in the inference time. On the top right, we can see that the pre-trained models achieve decent performance, but in the inference time, they incur a high computation cost. Interestingly, in the top-left region, we can see the IndoBERT-lite models, which achieve similar performance to the IndoBERT models, but with many fewer parameters and a slightly lower computation cost.

### 5.3 Multilingual vs. Monolingual Models

Based on Table 5, we can conclude that contextualized monolingual models outperform contextualized multilingual models on the classification tasks by a large margin, but on the sequence labeling tasks, multilingual models tend to perform better compared to monolingual models and even perform much better on the NERGrit and FacQA tasks. As shown in Appendix A, both the NERGrit and FacQA tasks contain many entity names which

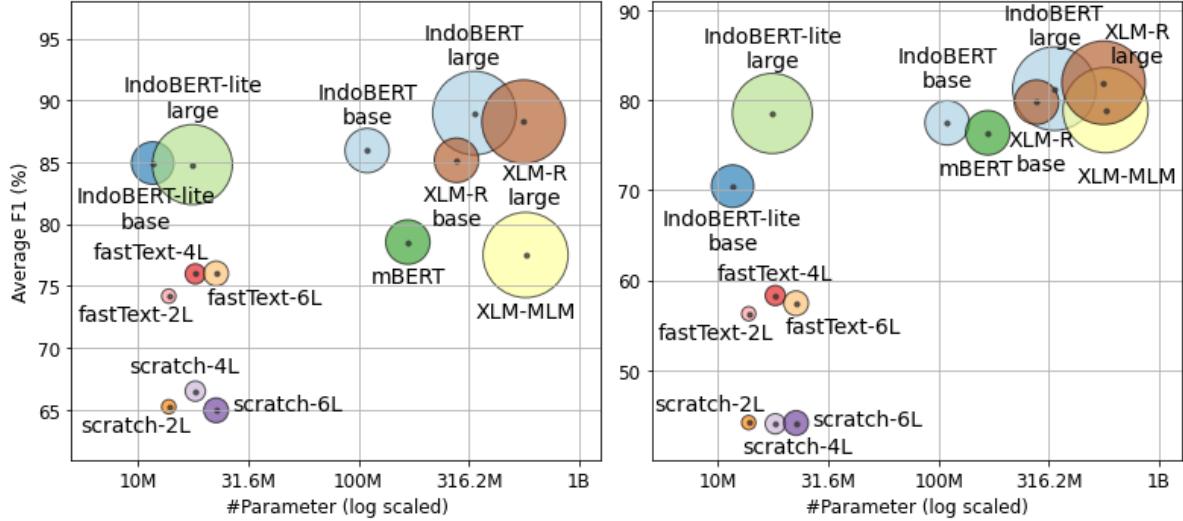


Figure 1: Performance-space trade-off for all baseline models on classification tasks (left) and sequence labeling tasks (right). We take the best model for each model size. 2L, 4L, and 6L denote the number of layers used in the model. The size of the dots represents the number of FLOPs of the model. We use python package `thop` taken from <https://pypi.org/project/thop/> to calculate the number of FLOPs.

come from other languages, especially English. These facts suggest that monolingual models capture the semantic meaning of a word better than multilingual models, but multilingual models identify foreign terms better than monolingual models.

#### 5.4 Effectiveness of Indo4B Dataset

Tasks	#Layer	fastText-cc-id	fastText-indo4b
Classification	2	72.00	<b>74.17</b>
	4	74.79	<b>75.97</b>
	6	74.80	<b>76.00</b>
Sequence Labeling	2	<b>56.26</b>	55.55
	4	57.97	<b>58.28</b>
	6	56.82	<b>57.42</b>

Table 6: Experiment results on fastText embeddings on IndoNLU tasks with different number of transformer layers

According to [Grave et al. \(2018\)](#), Common Crawl is a corpus containing over 24 TB.<sup>8</sup> We estimate the size of the CC-ID dataset to be around  $\approx 180$  GB uncompressed. Although the Indo4B dataset size is much smaller ( $\approx 23$  GB), Table 6 shows us that the fastText models trained on the Indo4B dataset (fastText-indo4b) consistently outperform fastText models trained on the CC-ID dataset (fastText-cc-id) in both classification and sequence labeling tasks in all model settings. Based

on Table 5, the fact that fastText-indo4b outperforms fastText-cc-id with a higher score on 10 out of 12 tasks suggests that a relatively smaller dataset ( $\approx 23$  GB) can significantly outperform its larger counterpart ( $\approx 180$  GB). We conclude that even though our Indo4B dataset is smaller, it covers more variety of the Indonesian language and has better text quality compared to the CC-ID dataset.

#### 5.5 Effectiveness of IndoBERT and IndoBERT-lite

Table 5 shows that the IndoBERT models outperform the multilingual models on 8 out of 12 tasks. In general, the IndoBERT models achieve the highest average score on the classification task. We conjecture that monolingual models learn better sentiment-level semantics on both colloquial and formal language styles than multilingual models, even though the IndoBERT models' size is 40%–60% smaller. On sequence labeling tasks, the IndoBERT models cannot perform as well as the multilingual models (XLM-R) in three sequence labeling tasks: POSP, NERGrit, and FacQA. One of the possible explanations is that these datasets have many borrowed words from English, and multilingual models have the advantage in transferring learning from English.

Meanwhile, the IndoBERT-lite models achieve a decent performance on both classification and sequence labeling tasks with the advantage of compact size. Interestingly, the IndoBERT-lite<sub>LARGE</sub>

<sup>8</sup><https://commoncrawl.github.io/cc-crawl-statistics/plots/languages>

model performance is on par with that of XLM-R<sub>BASE</sub> while having 16x fewer parameters. We also observe that increasing the maximum sequence length to 512 in phase two improves the performance on the sequence labeling tasks. Moreover, training the model with longer input sequences enables it to learn temporal information from a given text input.

## 6 Conclusion

We introduce the first Indonesian benchmark for natural language understanding, IndoNLU, which consists of 12 tasks, with different levels of difficulty, domains, and styles. To establish a strong baseline, we collect large clean Indonesian datasets into a dataset called Indo4B, which we use for training monolingual contextual pre-trained language models, called IndoBERT and IndoBERT-lite. We demonstrate the effectiveness of our dataset and our pre-trained models in capturing sentence-level semantics, and apply them to the classification and sequence labeling tasks. To help with the reproducibility of the benchmark, we release the pre-trained models, including the collected data and code. In order to accelerate the community engagement and benchmark transparency, we have set up a leaderboard website for the NLP community. We publish our leaderboard website at <https://indobenchmark.com/>.

## Acknowledgments

We want to thank Cahya Wirawan, Pallavi Jain, Irene Gianni, Martijn Wieriks, Ade Romadhony, and Andrea Madotto for insightful discussions about this project. We sincerely thank the three anonymous reviewers for their insightful comments on our paper.

## References

- Željko Agić and Ivan Vulić. 2019. Jw300: A wide-coverage parallel corpus for low-resource languages. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3204–3210.
- A. N. Azhar, M. L. Khodra, and A. P. Sutiono. 2019. Multi-label aspect categorization with convolutional neural networks and extreme gradient boosting. In *2019 International Conference on Electrical Engineering and Informatics (ICEEI)*, pages 35–40.
- Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.
- Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Ziqing Yang, Shijin Wang, and Guoping Hu. 2019. Pre-training with whole word masking for chinese bert.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.
- Arawinda Dinakaramani, Fam Rashel, Andry Luthfi, and Ruli Manurung. 2014. Designing an indonesian part of speech tagset and manually tagged indonesian corpus. In *2014 International Conference on Asian Language Processing, IALP 2014, Kuching, Malaysia, October 20-22, 2014*, pages 66–69. IEEE.
- Jordhy Fernando, Masayu Leylia Khodra, and Ali Akbar Septiandri. 2019. Aspect and opinion terms extraction using double embeddings and attention mechanism for indonesian hotel reviews.
- Filip Ginter, Jan Hajič, Juhani Luotolahti, Milan Straka, and Daniel Zeman. 2017. CoNLL 2017 shared task - automatically annotated raw texts and word embeddings. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.
- Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning word vectors for 157 languages. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.
- Devin Hoesen and Ayu Purwarianti. 2018. Investigating bi-lstm and crf with pos tag embedding for indonesian named entity tagger. In *2018 International Conference on Asian Language Processing (IALP)*, pages 35–38. IEEE.
- Arfinda Ilmania, Samuel Cahyawijaya, Ayu Purwarianti, et al. 2018. Aspect detection and sentiment classification using deep neural network for indonesian aspect-based sentiment analysis. In *2018 International Conference on Asian Language Processing (IALP)*, pages 62–67. IEEE.
- Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérvé Jégou, and Tomas Mikolov. 2016. Fasttext.zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*.

- Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.
- Kemal Kurniawan and Alham Fikri Aji. 2018. Toward a standardized and more accurate indonesian part-of-speech tagging. In *2018 International Conference on Asian Language Processing (IALP)*, pages 303–307. IEEE.
- Andrey Kutuzov and Elizaveta Kuzmenko. 2019. To lemmatize or not to lemmatize: How word normalisation affects ELMo performance in word sense disambiguation. In *Proceedings of the First NLP Workshop on Deep Learning for Natural Language Processing*, pages 22–28, Turku, Finland. Linköping University Electronic Press.
- Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *CoRR*, abs/1901.07291.
- Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.
- Hang Le, Loïc Vial, Jibril Frej, Vincent Segonne, Maximin Coavoux, Benjamin Lecouteux, Alexandre Alauzen, Benoît Crabbé, Laurent Besacier, and Didier Schwab. 2019. Flaubert: Unsupervised language model pre-training for french.
- Pierre Lison and Jörg Tiedemann. 2016. Opensubtitles2016: Extracting large parallel corpora from movie and tv subtitles. In *Proceedings of the 10th International Conference on Language Resources and Evaluation*.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.
- Miftahul Mahfuzh, Sidik Soleman, and Ayu Purwarianti. 2019. Improving joint layer rnn based keyphrase extraction by using syntactical features. In *2019 International Conference of Advanced Informatics: Concepts, Theory and Applications (ICAICTA)*, pages 1–6. IEEE.
- Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric Villemonte de la Clergerie, Djamel Seddah, and Benoît Sagot. 2019. Camembert: a tasty french language model.
- David Moeljadi. 2012. Usage of indonesian possessive verbal predicates: a statistical analysis based on questionnaire and storytelling surveys. In *APLL-5 conference. SOAS, University of London*.
- Hiroki Nomoto, Kenji Okano, David Moeljadi, and Hideo Sawada. 2018. Tufts asian language parallel corpus (talpco). In *Proceedings of the Twenty-fourth Annual Meeting of the Association for Natural Language Processing*, pages 436–439.
- Pedro Javier Ortiz Suárez, Benoît Sagot, and Laurent Romary. 2019. Asynchronous Pipeline for Processing Huge Corpora on Medium to Low Resource Infrastructures. In *7th Workshop on the Challenges in the Management of Large Corpora (CMLC-7)*, Cardiff, United Kingdom. Leibniz-Institut für Deutsche Sprache.
- Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237.
- Femphy Pisceldo, Rahmad Mahendra, Ruli Manurung, and I Wayan Arka. 2008. A two-level morphological analyser for the indonesian language. In *Proceedings of the Australasian Language Technology Association Workshop 2008*, pages 142–150.
- Ayu Purwarianti and Ida Ayu Putu Ari Crisdayanti. 2019. Improving bi-lstm performance for indonesian sentiment analysis using paragraph vector. In *2019 International Conference of Advanced Informatics: Concepts, Theory and Applications (ICAICTA)*, pages 1–5. IEEE.
- Ayu Purwarianti, Masatoshi Tsuchiya, and Seiichi Nakagawa. 2007. A machine learning approach for indonesian question answering system. In *Artificial Intelligence and Applications*, pages 573–578.
- Mei Silviana Saputri, Rahmad Mahendra, and Mirna Adriani. 2018. Emotion classification on indonesian twitter dataset. In *2018 International Conference on Asian Language Processing (IALP)*, pages 90–95. IEEE.
- Ali Akbar Septiandri and Arie Pratama Sutiono. 2019. Aspect and opinion term extraction for aspect based sentiment analysis of hotel reviews using transfer learning.
- Ken Nabila Setya and Rahmad Mahendra. 2018. Semi-supervised textual entailment on indonesian wikipedia data. In *2018 International Conference on Computational Linguistics and Intelligent Text Processing (CICLing)*.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, pages 3261–3275.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355.

Liang Xu, Xuanwei Zhang, Lu Li, Hai Hu, Chenjie Cao, Weitang Liu, Junyi Li, Yudong Li, Kai Sun, Yechen Xu, et al. 2020. Clue: A chinese language understanding evaluation benchmark. *arXiv preprint arXiv:2004.05986*.

## A Data Samples

In this section, we show examples for downstream tasks in the IndoNLU benchmark.

- The examples of SmSA task are shown in Table 7.
- The examples of EmoT task are shown in Table 8.
- The examples of KEPS task are shown in Table 9.
- The examples of HoASA task are shown in Table 10.
- The examples of CASA task are shown in Table 11.
- The examples of WReTE task are shown in Table 12.
- The examples of NERGrit task are shown in Table 13.
- The examples of NERP task are shown in Table 14.
- The examples of BaPOS task are shown in Table 15.
- The examples of POSP task are shown in Table 16.
- The examples of FacQA task are shown in Table 17.
- The examples of TermA task are shown in Table 18.

## B Indo4B Data Sources

In this section, we show the source of each dataset that we use to build our Indo4B dataset. The source of each corpus is shown in Table 19.

Sentence	Sentiment
pengecut dia itu , cuma bisa nantangin dari belakang saja	neg
wortel mengandung vitamin a yang bisa jaga kesehatan mata	neut
mocha float kfc itu minuman terenak yang pernah gue rasain	pos

Table 7: Sample data on task SmSA

Tweet	Emotion
Masalah ga akan pernah menjauh, hadapi Selasamu dengan penuh semangat!	happy
Sayang seribu sayang namun tak ada satupun yg nyangkut sampai sekarang	sadness
cewek suka bola itu dimata cowok cantiknya nambah, biarpun matanya panda	love

Table 8: Sample data on task EmoT

## C Pre-Training Hyperparameters

In this section, we show all hyperparameters used in our IndoBERT and IndoBERT-lite training process. The hyperparameters is shown in Table 20.

## D Fine-Tuning Hyperparameters

In this section, we show all hyperparameters used in the fine-tuning process of each baseline model. The hyperparameter configuration is shown in Table 21.

## E Extensive Experiment Results on IndoNLU Benchmark

In this section, we show all experiments conducted in the IndoNLU benchmark. We use a batch size of 16 for all datasets except FacQA and POSP, for which we use a batch size of 8. The results of the full experiments are shown in Table 22.

Word Keyphrase	Layanan O	BCA B	Mobile I	Banking I	Bermasalah B
Word Keyphrase	Tidak O	mengecewakan O	pakai B	BCA B	Mobile I
Word Keyphrase	nggak B	ada I	tandingannya I	e-channel B	BCA I

Table 9: Sample data on task KEPS

Sentence	Aspect										
	AC	Air Panas	Bau	General	Kebersihan	Linen	Service	Sunrise Meal	TV	WiFi	
air panas kurang berfungsi dan handuk lembab.	neut	neg	neut	neut	neut	neg	neut	neut	neut	neut	neut
Shower zonk, resepsionis yang wanita judes	neut	neut	neut	neut	neut	neut	neg	neut	neut	neut	neut
Kamar kurang bersih, terutama kamar mandi.	neut	neut	neut	neut	neg	neut	neut	neut	neut	neut	neut

Table 10: Sample data on task HoASA

Sentence	Aspect					
	Fuel	Machine	Others	Part	Price	Service
bodi plus tampilan nya Avanza baru mantap juragan	neut	neut	neut	pos	neut	neut
udah gaya nya stylish ekonomis pula, beli calya deh	neut	neut	neut	pos	pos	neut
Mobil kualitas jelek kayak wuling saja masuk Indonesia	neut	neut	neg	neut	neut	neut

Table 11: Sample data on task CASA

Sentence A	Sentence B	Label
Anak sebaiknya menjalani tirah baring	Anak sebaiknya menjalani istirahat	Entail or Paraphrase
Kedua kata ini ditulis dengan huruf kanji yang sama	Jepang disebut Nippon atau Nihon dalam bahasa Jepang	Not Entail
Elektron hanya menduduki 0,06% massa total atom	Elektron hanya mengambil 0,06% massa total atom	Entail or Paraphrase

Table 12: Sample data on task WReTE

Word Entity	Produser O	David B-PER	Heyman I-PER	dan O	sutradara O	Mark B-PER	Herman I-PERS	sedang O	mencari O	seseorang O
Word Entity	Pada O	tahun O	1996 O	Williams B-PER	pindah O	ke O	Sebastopol B-PLA	,	California O	di B-PLA
Word Entity	bekerja O	untuk O	penerbitan O	perusahaan O	teknologi O	O	B-ORG	,	Reilly I-ORG	Media I-ORG

Table 13: Sample data on task NERGrit. PER = PERSON, ORG = ORGANIZATION, PLA = PLACE

Word Entity	kepala O	dinas O	tata O	kota O	manado B-PLC	amos B-PPL	kenda I-PPL	menyatakan O	tidak O	tahu O
Word Entity	telah O	mendaftar O	untuk O	menjadi O	official O	merchant O	bandung B-EVT	great I-EVT	sale I-EVT	2017 I-EVT
Word Entity	sekitar O	timur B-PLC	dan O	barat B-PLC	arnhem I-PLC	,	O B-PLC	katherine B-PLC	dan O	daerah O

Table 14: Sample data on task NERP. PLC = PLACE, PPL = PEOPLE, EVT = EVENT

Word Tag	Pemerintah B>NNP	kota B>NNP	Delhi B>NNP	mengerahkan B-VB	monyet B-NN	untuk B-SC	mengusir B-VB	monyet-monyet B-NN	lain B-JJ	yang B-SC
Word Tag	Beberapa B-CD	laporan B-NN	menyebutkan B-VB	setidaknya B-RB	10 B-CD	monyet B-NN	ditempatkan B-VB	di B-IN	luar B-NN	arena B-NN
Word Tag	berencana B-VB	mendatangkan B-CD	10 B-CD	monyet B-NN	sejenis B-NN	dari B-IN	negara B-NNP	bagian I-NNP	Rajasthan B-NNP	.

Table 15: Sample data on task BaPOS. POS tag labels follow Universitas Indonesia POS Tag Standard.<sup>9</sup>

Word Tag	kepala B-NNO	dinas B-VBP	tata B-NNO	kota B-NNO	manado B-NNP	amos B-NNP	kenda B-NNP	menyatakan B-VBT	tidak B-NEG	tahu B-VBI
Word Tag	telah B-ADK	mendaftar B-VBI	untuk B-PPO	menjadi B-VBL	official B-NNO	merchant B-NNP	bandung B-NNP	great B-NNP	sale B-NNP	2017 B-NUM
Word Tag	sekitar B-PPO	timur B-NNP	dan B-CCN	barat B-NNP	arnhem B-NNP	,	B-SYM B-NNP	katherine B-CCN	daerah B-NNO	sekitar B-ADV

Table 16: Sample data on task POSP POS tag labels follow INACL POS Tagging Convention.<sup>10</sup>

<b>Question</b>	"Siapakah penasihat utama Presiden AS George W Bush?"						
<b>Passage Label</b>	Nasib O	Karl B	Rove I	Akan O	Segera O	Diputuskan O	
<b>Question</b>	"Dimana terjadinya letusan gunung berapi dahsyat tahun 1883?"						
<b>Passage Label</b>	Di O	Kepulauan B	Krakatau I	Terdapat O	400 O	Tanaman O	
<b>Question</b>	"Perusahaan apakah yang sejak 1 Januari 2006, menurunkan harga pertamax dan pertamax plus?"						
<b>Passage Label</b>	Pesaing O	Semakin O	Banyak O	,	Pertamina B	Berusaha O	Kompetitif O

Table 17: Sample data on task FacQA

<b>Word Entity</b>	sayang O	wifi B-ASP	tidak B-SEN	bagus I-SEN	harus O	keluar O	kamar O	.	fasilitas O	lengkap B-ASP
<b>Word Entity</b>	pelayanan B-ASP	nya I-ASP	sangat B-SEN	bagus I-SEN	.	kamar O	nya I-ASP	juga O	oke B-SEN	.
<b>Word Entity</b>	kamar B-ASP	cukup B-SEN	luas I-SEN	,	interior O	menarik B-ASP	dan B-SEN	unik O	sekali I-SEN	,

Table 18: Sample data on task TermA. SEN = SENTIMENT, ASP = ASPECT

Corpus Name	Source	Public URL
OSCAR	OSCAR	<a href="https://oscar-public.huma-num.fr/compressed/id_dedup.txt.gz">https://oscar-public.huma-num.fr/compressed/id_dedup.txt.gz</a>
CoNLLu Common Crawl	LINDAT/CLARIAH-CZ	<a href="https://lindat.mff.cuni.cz/repository/xmlui/bitstream/handle/11234/1-1989/Indonesian-annotated-conll17.tar">https://lindat.mff.cuni.cz/repository/xmlui/bitstream/handle/11234/1-1989/Indonesian-annotated-conll17.tar</a>
OpenSubtitles	OPUS OpenSubtitles	<a href="http://opus.nlpl.eu/download.php?f=OpenSubtitles/v2016/mono/OpenSubtitles.raw.id.gz">http://opus.nlpl.eu/download.php?f=OpenSubtitles/v2016/mono/OpenSubtitles.raw.id.gz</a>
Wikipedia Dump	Wikipedia	<a href="https://dumps.wikimedia.org/idwiki/20200401/idwiki-20200401-pages-articles-multistream.xml.bz2">https://dumps.wikimedia.org/idwiki/20200401/idwiki-20200401-pages-articles-multistream.xml.bz2</a>
Wikipedia CoNLLu	LINDAT/CLARIAH-CZ	<a href="https://lindat.mff.cuni.cz/repository/xmlui/bitstream/handle/11234/1-1989/Indonesian-annotated-conll17.tar">https://lindat.mff.cuni.cz/repository/xmlui/bitstream/handle/11234/1-1989/Indonesian-annotated-conll17.tar</a>
Twitter Crawl	Twitter	Not publicly available
Twitter UI	Twitter	Not publicly available
OPUS JW300	OPUS	<a href="http://opus.nlpl.eu/JW300.php">http://opus.nlpl.eu/JW300.php</a>
Tempo	ILSP	<a href="http://ilps.science.uva.nl/ilps/wp-content/uploads/sites/6/files/bahasaindonesia/tempo.zip">http://ilps.science.uva.nl/ilps/wp-content/uploads/sites/6/files/bahasaindonesia/tempo.zip</a>
Kompas	ILSP	<a href="http://ilps.science.uva.nl/ilps/wp-content/uploads/sites/6/files/bahasaindonesia/kompas.zip">http://ilps.science.uva.nl/ilps/wp-content/uploads/sites/6/files/bahasaindonesia/kompas.zip</a>
TED	TED	<a href="https://github.com/ajinkyakulkarni14/TED-Multilingual-Parallel-Corpus/tree/master/Monolingual_data">https://github.com/ajinkyakulkarni14/TED-Multilingual-Parallel-Corpus/tree/master/Monolingual_data</a>
BPPT	BPPT	<a href="http://www.pan10n.net/english/outputs/Indonesia/BPPT/0902/BPPTIndToEngCorpusHalfM.zip">http://www.pan10n.net/english/outputs/Indonesia/BPPT/0902/BPPTIndToEngCorpusHalfM.zip</a>
Parallel Corpus	PAN Localization	<a href="http://pan10n.net/english/outputs/Indonesia/UI/0802/Parallel%20Corpus.zip">http://pan10n.net/english/outputs/Indonesia/UI/0802/Parallel%20Corpus.zip</a>
TALPCo	Tokyo University	<a href="https://github.com/mathbaha/TALPCo">https://github.com/mathbaha/TALPCo</a>
Frog Storytelling	Tokyo University	<a href="https://github.com/davidmoeljadi/corpus-frog-storytelling">https://github.com/davidmoeljadi/corpus-frog-storytelling</a>

Table 19: Indo4B Corpus

Hyperparameter	IndoBERT <sub>BASE</sub>	IndoBERT <sub>LARGE</sub>	IndoBERT-lite <sub>BASE</sub>	IndoBERT-lite <sub>LARGE</sub>
attention_probs_dropout_prob	0.1	0.1	0	0
hidden_act	gelu	gelu	gelu	gelu
hidden_dropout_prob	0.1	0.1	0	0
embedding_size	768	1024	128	128
hidden_size	768	1024	768	1024
initializer_range	0.02	0.02	0.02	0.02
intermediate_size	3072	4096	3072	4096
max_position_embeddings	512	512	512	512
num_attention_heads	12	16	12	16
num_hidden_layers	12	24	12	24
type_vocab_size	2	2	2	2
vocab_size	30522	30522	30000	30000
num_hidden_groups	-	-	1	1
net_structure_type	-	-	0	0
gap_size	-	-	0	0
num_memory_blocks	-	-	0	0
inner_group_num	-	-	1	1
down_scale_factor	-	-	1	1

Table 20: Hyperparameter configurations for IndoBERT and IndoBERT-lite pre-trained models.

	<b>batch_size</b>	<b>n_layers</b>	<b>n_epochs</b>	<b>lr</b>	<b>early_stop</b>	<b>gamma</b>	<b>max_norm</b>	<b>seed</b>
Scratch	[8,16]	[2,4,6]	25	1e-4	12	0.9	10	42
fastText-cc-id	[8,16]	[2,4,6]	25	1e-4	12	0.9	10	42
fastText-indo4B	[8,16]	[2,4,6]	25	1e-4	12	0.9	10	42
mBERT	[8,16]	12	25	1e-5	12	0.9	10	42
XLM-MLM	[8,16]	16	25	1e-5	12	0.9	10	42
XLM-R <sub>BASE</sub>	[8,16]	12	25	2e-5	12	0.9	10	42
XLM-R <sub>LARGE</sub>	[8,16]	24	25	1e-5	12	0.9	10	42
IndoBERT-lite <sub>BASE</sub>	[8,16]	12	25	1e-5	12	0.9	10	42
+ phase 2	[8,16]	12	25	1e-5	12	0.9	10	42
IndoBERT-lite <sub>LARGE</sub>	[8,16]	24	25	[1e-5,2e-5]	12	0.9	10	42
+ phase 2	[8,16]	24	25	2e-5	12	0.9	10	42
IndoBERT <sub>BASE</sub>	[8,16]	12	25	[1e-5,4e-5]	12	0.9	10	42
+ phase 2	[8,16]	12	25	4e-5	12	0.9	10	42
IndoBERT <sub>LARGE</sub>	[8,16]	24	25	4e-5	12	0.9	10	42
+ phase 2	[8,16]	24	25	[3e-5,4e-5]	12	0.9	10	42

Table 21: Hyperparameter configurations for fine-tuning in IndoNLU benchmark. We use a batch size of 8 for POSP and FacQA, and a batch size of 16 for EmoT, SmSA, CASA, HoASA, WReTE, BaPOS, TermA, KEPS, NERGrit, and NERP.

Model	LR	# Layer	Param	Classification						Sequence Labeling							
				EmoT	SmSA	CASA	HoASA	WReTE	Avg	POSP	BaPOS	TermA	KEPS	NERGrit	NERP	FacQA	Avg
scratch	1e-4	2	38.6M	58.51	64.22	65.58	78.31	59.54	65.23	85.69	66.30	69.67	47.71	4.62	31.14	4.08	44.17
scratch	1e-4	4	52.8M	57.31	67.35	67.15	76.28	64.35	66.49	86.78	70.24	70.36	39.40	5.80	30.66	5.00	44.03
scratch	1e-4	6	67.0M	52.84	67.07	69.88	76.83	58.06	64.94	86.16	68.18	70.64	45.65	5.14	27.88	5.21	44.12
fasttext-cc-id-300-no-oov-uncased	1e-4	6	15.1M	67.43	78.84	81.61	85.01	61.13	74.80	94.36	78.45	77.26	57.28	26.70	46.36	17.3	56.82
fasttext-cc-id-300-no-oov-uncased	1e-4	4	10.7M	65.36	76.92	79.02	85.32	67.36	74.79	94.35	79.85	76.12	56.39	37.32	46.46	15.29	57.97
fasttext-cc-id-300-no-oov-uncased	1e-4	2	6.3M	64.74	76.71	75.39	78.05	65.11	72.00	94.42	78.12	73.45	55.22	33.27	45.44	13.89	56.26
fasttext-4B-id-300-no-oov-uncased	1e-4	6	15.1M	68.47	83.07	81.96	86.20	60.33	76.00	95.15	80.61	75.26	44.71	40.83	47.02	18.39	57.42
fasttext-4B-id-300-no-oov-uncased	1e-4	4	10.7M	69.23	82.13	82.20	85.88	60.42	75.97	94.94	81.77	74.43	56.70	38.69	46.79	14.65	58.28
fasttext-4B-id-300-no-oov-uncased	1e-4	2	6.3M	70.97	83.63	78.97	80.16	57.11	74.17	94.93	80.11	71.92	56.67	31.46	45.08	8.65	55.55
indobert-lite-base-128-112.5k	1e-5	12	11.7M	73.88	90.85	89.68	88.07	82.17	84.93	91.40	75.10	89.29	69.02	66.62	46.58	54.99	70.43
indobert-lite-base-128-191.5k	1e-5	12	11.7M	71.95	89.87	84.71	87.57	80.30	82.88	87.27	67.33	89.15	65.84	67.67	49.32	51.76	68.33
indobert-lite-base-512-162.5k	1e-5	12	11.7M	72.27	90.29	87.63	87.62	83.62	84.29	90.05	77.59	89.19	69.13	66.71	50.52	49.18	70.34
indobert-base-128	4e-5	12	124.5M	75.48	87.73	93.23	92.07	78.55	85.41	95.26	87.09	90.73	70.36	69.87	75.52	53.45	77.47
indobert-base-512	1e-5	12	124.5M	76.61	90.90	91.77	90.70	79.73	85.94	95.10	86.25	90.58	69.39	63.67	75.36	53.14	76.21
indobert-base-512	4e-5	12	124.5M	76.28	87.66	93.24	92.70	78.68	85.71	95.23	85.72	91.13	69.17	67.42	75.68	57.06	77.34
indobert-lite-large-128	1e-5	24	17.7M	75.19	88.66	90.99	89.53	78.98	84.67	91.56	83.74	90.23	67.89	71.19	74.37	65.50	77.78
indobert-lite-large-128	1e-5	24	17.7M	71.67	90.13	88.88	88.80	81.19	84.13	91.53	83.51	90.07	67.36	73.27	74.34	69.47	78.51
indobert-lite-large-128	2e-5	24	17.7M	70.80	88.61	88.13	91.05	85.41	84.80	94.53	84.91	90.72	68.55	73.07	74.89	62.87	78.51
indobert-large-128-1100k	4e-5	24	335.2M	77.04	93.71	96.64	93.27	84.17	88.97	95.71	89.74	91.97	70.82	70.76	77.54	67.27	80.55
indobert-large-128-1000k	4e-5	24	335.2M	77.08	92.72	95.69	93.75	82.91	88.43	95.71	90.35	91.87	71.18	77.60	79.25	62.48	81.21
indobert-large-512-1100k	4e-5	24	335.2M	77.39	92.90	95.90	93.77	81.62	88.32	95.25	86.05	91.92	69.71	75.20	77.53	69.86	80.79
indobert-large-512-1100k	3e-5	24	335.2M	79.47	92.03	94.94	93.38	80.30	88.02	95.34	87.36	92.14	71.27	76.63	77.99	68.09	81.26
bert-base-multilingual-uncased	1e-5	12	167.4M	67.30	84.14	72.23	84.63	84.40	78.54	91.85	83.25	89.51	64.31	75.02	69.27	61.29	76.36
xlm-mlm-100-1280	1e-5	16	573.2M	65.75	86.33	82.17	88.89	64.35	77.50	95.87	88.40	90.55	65.35	74.75	75.06	62.15	78.88
xlm-roberta-base	2e-5	12	278.7M	71.15	91.39	91.71	91.57	79.95	85.15	95.16	84.64	90.99	68.82	79.09	75.03	64.58	79.76
xlm-roberta-large	1e-5	24	561.0M	78.51	92.35	92.40	94.27	83.82	88.27	92.73	87.03	91.45	70.88	78.26	78.52	74.61	81.92

Table 22: Results of all experiments conducted in IndoNLU benchmark. We sample each batch with a size of 16 for all datasets except FacQA and POSP, for which we use a batch size of 8.

# Happy Are Those Who Grade without Seeing: A Multi-Task Learning Approach to Grade Essays Using Gaze Behaviour

Sandeep Mathias<sup>♦</sup>, Rudra Murthy<sup>♦,◊</sup>, Diptesh Kanodia<sup>♦,‡</sup>, Abhijit Mishra<sup>◊</sup>, Pushpak Bhattacharyya<sup>♦</sup>

<sup>♦</sup> Department of Computer Science, Indian Institute of Technology, Bombay

<sup>◊</sup> IBM Research, India

<sup>‡</sup> IITB-Monash Research Academy

{sam, rudra, diptesh, pb}@cse.iitb.ac.in, abhijitmishra.530@gmail.com

## Abstract

The gaze behaviour of a reader is helpful in solving several NLP tasks such as automatic essay grading. However, collecting gaze behaviour from readers is costly in terms of time and money. In this paper, we propose a way to improve automatic essay grading using gaze behaviour, which is learnt at run time using a multi-task learning framework. To demonstrate the efficacy of this multi-task learning based approach to automatic essay grading, we collect gaze behaviour for 48 essays across 4 essay sets, and learn gaze behaviour for the rest of the essays, numbering over 7000 essays. Using the learnt gaze behaviour, we can achieve a statistically significant improvement in performance over the state-of-the-art system for the essay sets where we have gaze data. We also achieve a statistically significant improvement for 4 other essay sets, numbering about 6000 essays, where we have no gaze behaviour data available. Our approach establishes that learning gaze behaviour improves automatic essay grading.

## 1 Introduction

Collecting a reader's psychological input can be very beneficial to a number of Natural Language Processing (NLP) tasks, like complexity (Mishra et al., 2017; González-Garduño and Søgaard, 2017), sentence simplification (Klerke et al., 2016), text understanding (Mishra et al., 2016), text quality (Mathias et al., 2018), parsing (Hale et al., 2018), etc. This psychological information can be extracted using devices like eye-trackers, and electroencephalogram (EEG) machines. However, one of the challenges in using reader's information involves collecting the psycholinguistic data itself.

In this paper, we choose the task of automatic essay grading and show how we can predict the score that a human rater would give using both text and *learnt* gaze behaviour. An essay is a piece of

text, written in response to a topic, called a prompt. Automatic essay grading is assigning a score to the essay using a machine. An essay set is a set of essays written in response to the same prompt.

Multi-task learning (Caruana, 1998) is a machine learning paradigm where we utilize auxiliary tasks to aid in solving a primary task. This is done by exploiting similarities between the primary task and the auxiliary tasks. **Scoring the essay** is the *primary task* and **learning gaze behaviour** is the *auxiliary task*.

Using gaze behaviour for a very small number of essays (**less than 0.7% of the essays in an essay set**), we see an improvement in predicting the overall score of the essays. We also use our gaze behaviour dataset to run experiments on **unseen** essay sets - *i.e.*, essay sets which have **no gaze behaviour data** - and observe improvements in the system's performance in automatically grading essays.

**Contributions** The main contribution of our paper is describing how we use gaze behaviour information, in a multi-task learning framework, to automatically score essays outperforming the state-of-the-art systems. We will also release the gaze behaviour dataset<sup>1</sup> and code<sup>2</sup> - the first of its kind, for automatic essay grading - to facilitate further research in using gaze behaviour for automatic essay grading and other similar NLP tasks.

### 1.1 Gaze Behaviour Terminology

An **Interest Area** (IA) is an area of the screen that we are interested in. These areas are where some text is displayed, and not the white background on the left/right, as well as above/below the text. **Each word** is a separate and unique IA.

<sup>1</sup>Gaze behaviour dataset: <http://www.cfilt.iitb.ac.in/cognitive-nlp/>  
Essays: <https://www.kaggle.com/c/asap-aes>

<sup>2</sup><https://github.com/lwsam/ASAP-Gaze>

A **Fixation** is an event when the reader’s eye is focused on a part of the screen. For our experiments, we are concerned only with fixations that occur within the interest areas. Fixations that occur in the background are ignored.

A **Saccade** is the path of the eye movement, as it goes from one fixation to the next. There are two types of saccades - Progressions and Regressions. **Progressions** are saccades where the reader moves from the current interest area to a *later* one. **Regressions** are saccades where the reader moves from the current interest area to an *earlier* one.

The rest of the paper is organized as follows. Section 2 describes our motivation for using eye-tracking and learning gaze behaviour from readers, over unseen texts. Section 3 describes some of the related work in the area of automatic essay grading, eye tracking and multi-task learning. Section 4 describes the gaze behaviour attributes used in our experiments, and the intuition behind them. We describe our dataset creation and experiment setup in Section 5. In Section 6, we report our results and present a detailed analysis. We present our conclusions and discuss possible future work in Section 7.

## 2 Motivation

Mishra and Bhattacharyya (2018), for instance, describe a lot of research in solving multiple problems in NLP using gaze behaviour of readers. However, most of their work involves collecting the gaze behaviour data first, and then splitting the data into training and testing data, before performing their experiments. While their work did show significant improvements over baseline approaches, across multiple NLP tasks, collecting the gaze behaviour data would be quite expensive, both in terms of time and money.

Therefore, we ask ourselves: “*Can we learn gaze behaviour, using a small amount of seed data, to help solve an NLP task?*” In order to use gaze behaviour on a large scale, we need to be able to *learn* it, since we can not ask a user to read texts every time we wish to use gaze behaviour data. Mathias et al. (2018) describe using gaze behaviour to predict how a reader would rate a piece of text (which is similar to our chosen application). Since they showed that gaze behaviour can help in predicting text quality, we use multi-task learning to simultaneously learn gaze behaviour information (auxiliary task) as well as score the essay (the

primary task). However, they **collect all their gaze behaviour data *a priori***, while *we try to learn the gaze behaviour of a reader* and use what we learn from our system, for grading the essays. Hence, while they showed that gaze behaviour *could* help in predicting how a reader would score a text, their approach requires a reader to **read the text**, while our approach does not do so, *during testing / deployment*.

## 3 Related Work

### 3.1 Automatic Essay Grading (AEG)

The very first AEG system was proposed by Page (1966). Since then, there have been a lot of other AEG systems (see Shermis and Burstein (2013) for more details). In 2012, the Hewlett Foundation released a dataset called the Automatic Student Assessment Prize (ASAP) AEG dataset. The dataset contains about 13,000 essays across eight different essay sets. We discuss more about that dataset later.

With the availability of a large dataset, there has been a lot of research, especially using neural networks, in automatically grading essays - like using Long Short Term Memory (LSTM) Networks (Taghipour and Ng, 2016; Tay et al., 2018), Convolutional Neural Networks (CNNs) (Dong and Zhang, 2016), or both (Dong et al., 2017). Zhang and Litman (2018) improve on the results of Dong et al. (2017) using co-attention between the source article and the essay for one of the types of essay sets.

### 3.2 Eye-Tracking

Capturing the gaze behaviour of readers has been found to be quite useful in improving the performance of NLP tasks (Mishra and Bhattacharyya, 2018). The main idea behind using gaze behaviour is the eye-mind hypothesis (Just and Carpenter, 1980), which states that whatever text the eye reads, that is what the mind processes. This hypothesis has led to a large body of work in psycholinguistic research that shows a relationship between text processing and gaze behaviour. Mishra and Bhattacharyya (2018) also describe some of the ways that eye-tracking can be used for multiple NLP tasks like translation complexity, sentiment analysis, etc.

Research has been done on using gaze behaviour at run time to solve downstream NLP tasks like sentence simplification (Klerke et al., 2016), readability (González-Garduño and Søgaard, 2018; Singh

et al., 2016), part-of-speech tagging (Barrett et al., 2016), sentiment analysis (Mishra et al., 2018; Barrett et al., 2018; Long et al., 2019), grammatical error detection (Barrett et al., 2018), hate speech detection (Barrett et al., 2018) and named entity recognition (Hollenstein and Zhang, 2019).

Different strategies have been adopted to alleviate the need for gaze behaviour at run time. Barrett et al. (2016) use token level averages of gaze features at run time from the Dundee Corpus (Kennedy et al., 2003), to alleviate the need for gaze behaviour at run time. Singh et al. (2016) and Long et al. (2019) predict gaze behaviour at the token-level prior to using it at run time. Mishra et al. (2018), González-Garduño and Søgaard (2018), Barrett et al. (2018), and Klerke et al. (2016), use multi-task learning to learn gaze behaviour along with solving the primary NLP task.

## 4 Gaze Behaviour Attributes

In our experiments, we use only a subset of gaze behaviour attributes described by Mathias et al. (2018) because most of the other attributes (like Second Fixation Duration<sup>3</sup>) were mostly 0, for most of the interest areas, and learning over them would not have yielded any meaningful results.

**Fixation Based Attributes** In our experiments, we use the **Dwell Time** (DT) and **First Fixation Duration** (FFD) as fixation-based gaze behaviour attributes. Dwell Time is the total amount of time a user spends focusing on an interest area. First Fixation Duration is amount of time that a reader initially focuses on an interest area. Larger values for fixation durations (for both DT and FFD) usually indicate that a word could be wrong (either a spelling mistake or grammar error). Errors would force a reader to pause, as they try to understand why the error was made (For example, if the writer wrote “short *cat*” instead of “short *cut*”).

**Saccade Based Attribute** In addition to the Fixation based attributes, we also look at a regression-based attribute - **IsRegression** (IR). This attribute is used to check whether or not a regression occurred from a given interest area. We don’t focus on progression-based attributes, because the usual direction of reading is progressions. We are mainly concerned with regressions because they often occur when there is a mistake, or a need for disam-

<sup>3</sup>The duration of the fixation when the reader fixates on an interest area for the second time.

biguation (like trying to resolve the antecedent of an anaphora).

**Interest Area Based Attributes** Lastly, we also use IA-based attributes, such as the **Run Count** (RC) and if the IA was **Skipped** (Skip). The Run Count is the number of times a particular IA was fixated on, and Skip is whether or not the IA was skipped. A well-written text would be read more easily, meaning a lower RC, and higher Skip (Mathias et al., 2018).

## 5 Dataset and Experiment Setup

### 5.1 Essay Dataset Details

We perform our experiments on the ASAP AEG dataset. The dataset has approximately 13,000 essays, across 8 essay sets. Table 1 reports the statistics of the dataset in terms of Number of Essays, Score Range, and Mean Word Count. The first 4 rows in Table 1 are **source-dependent response** (SDR) essay sets, which we use to collect our gaze behaviour data. The other essays are used as **unseen** essay sets. **SDRs** are essays written in response to a question about a source article. For example, one of the essay sets that we use is based on an article called *The Mooring Mast*, by Marcia Amidon Lusted<sup>4</sup>.

### 5.2 Evaluation Metric

Essay Set	Number of Essays	Score Range	Mean Word Count
Prompt 3	1726	0-3	150
Prompt 4	1770	0-3	150
Prompt 5	1805	0-4	150
Prompt 6	1800	0-4	150
Prompt 1	1783	2-12	350
Prompt 2	1800	1-6	350
Prompt 7	1569	0-30	250
Prompt 8	723	0-60	650
<b>Total</b>	12976	0-60	250

Table 1: Statistics of the 8 essay sets from the ASAP AEG dataset. We collect gaze behaviour data **only for Prompts 3 - 6**, as explained in Section 5.3. The other 4 prompts comprise our **unseen** essay sets.

For measuring our system’s performance, we use Cohen’s Kappa with quadratic weights - Quadratic Weighted Kappa (QWK) (Cohen, 1968) for the following reasons. Firstly, irrespective of whether we

<sup>4</sup>The prompt is “Based on the excerpt, describe the obstacles the builders of the Empire State Building faced in attempting to allow dirigibles to dock there. Support your answer with relevant and specific information from the excerpt.” The original article is present in Appendix A.

use regression, or ordinal classification, the final scores that are predicted by the system should be discrete scores. Hence, using Pearson Correlation would not be appropriate for our system. Secondly, F-Score and accuracy do not consider chance agreements unlike Cohen’s Kappa. If we were to give everyone an average grade, we would get a positive value for accuracy and F-Score, but a Kappa value of 0. Thirdly, *weighted* Kappa takes into account the fact that the classes are ordered, i.e.  $0 < 1 < 2 \dots$ . Using unweighted Kappa would penalize a 0 graded as a 4, as much as a 1. We use quadratic weights, as opposed to linear weights, because quadratic weights reward agreements and penalize mismatches more than linear weights.

### 5.3 Creation of the Gaze Behaviour Dataset

In this subsection, we describe how we created our gaze behaviour dataset, how we chose our essays for eye-tracking, and how they were annotated.

#### 5.3.1 Details of Texts

Essay Set	0	1	2	3	4	Total
Prompt 3	2	4	5	1	N/A	12
Prompt 4	2	3	4	3	N/A	12
Prompt 5	2	1	3	5	1	12
Prompt 6	2	2	3	4	1	12
<b>Total</b>	8	10	15	13	2	48

Table 2: Number of essays for each essay set which we collected gaze behaviour, scored between 0 to 3 (or 4).

As mentioned earlier in Section 5, we used only essays corresponding to prompts 3 to 6 of the ASAP AEG dataset. From **each of the four essay sets**, we selected **12 essays** with a diverse vocabulary as well as all possible scores.

We use a greedy algorithm to select essays *i.e.*, For each essay set, we pick 12 essays, covering all score points with maximum number of unique tokens, as well as being under 250 words. Table 2 reports the distribution of essays with each score, for each of the 4 essay sets that we use to create our gaze behaviour dataset.

To display the essay text on the screen, we use a large font size, so that (a) the text is clear, and (b) the reader’s gaze is captured on the words which they are currently reading. Although, this ensures the clarity in reading and recording the gaze pattern in a more accurate manner, it also imposes a limitation on the size of the essay which can be used for

our experiment. This is why, the longest essay in our gaze behaviour dataset is **about 250 words**.

The original essays have their named entities anonymized. Hence, before running the experiments, we replaced the required named entities with placeholders (Eg. @NAME1 → “Al Smith”, @PLACE1 → “New Jersey”, @MONTH1 → “May”, etc.)<sup>5</sup>.

#### 5.3.2 Annotator Details

We used a total of **8 annotators, aged between 18 and 31**, with an **average age of 25 years**. All of them were either in college, or had completed a Bachelor’s degree. All but one of them also had experience as a teaching assistant. The annotators were fluent in English, and about half of them had participated earlier, in similar experiments. The annotators were adequately compensated for their work<sup>6</sup>.

To assess the quality of the individual annotators, we evaluated the scores they provided against the ground truth scores - *i.e.*, the scores given by the original annotators. The **QWK** measures the agreement between the annotators and the ground truth score. **Close** is the number of times (out of 48) in which the annotators either agreed with the ground truth scores, or differed from them by **at most 1** score point. **Correct** is the number of times (out of 48) in which the annotators agreed with the ground truth scores. The mean values for the 3 measures were **0.646** (QWK), **42.75** (Close) and **22.25** (Correct).

#### 5.4 System Details

We conduct our experiments using well-established norms in eye-tracking research ([Holmqvist et al., 2011](#)). The essays are displayed on a screen that is kept about **2 feet** in front of the participant.

The workflow of the experiment is as follows. First, **the camera is calibrated**. This is done by having the annotator look at **13 points** on the screen, while the camera tracks their eyes. Next, **the calibration is validated**. In this step, the participant looks at the same points they saw earlier. If there is a big difference between the participant’s fixation points tracked by the camera and the actual points, calibration is repeated. Then, **the reader**

<sup>5</sup>Another advantage of using source-dependent essays is that there is a source article which we can use to correctly replace the anonymized named entities

<sup>6</sup>We report details on individual annotators in Appendix B.

**performs a self-paced reading of the essay** while we supervise the tracking of their eyes. After reading and scoring an essay, the participant takes a small break of **about a minute**, before continuing. Before the next essay is read, the camera has to again be calibrated and validated<sup>7</sup>. The essay is displayed on the screen in **Times New Roman** typeface with a **font size of 23**. Finally, **the reader scores the essay and provides a justification for their score**<sup>8</sup>.

This entire process is done using an **SR Research Eye Link 1000** eye-tracker (monocular stabilized head mode, with a sampling rate of 500Hz). The machine collects all the gaze details that we need for our experiments. An interest area report is generated for gaze behaviour using the **SR Research Data Viewer** software.

## 5.5 Experiment Details

We use **five-fold cross-validation** to evaluate our system. For each fold, **60%** is used as training, **20%** for validation, and **20%** for testing. The folds are the same as those used by [Taghipour and Ng \(2016\)](#). Prior to running our experiments, we convert the scores from their original score range (given in Table 1) to the **range of [0, 1]** as described by [Taghipour and Ng \(2016\)](#).

In order to normalize idiosyncratic reading patterns across different readers, we perform binning for each of the features for each of the readers. For IR and Skip we use only two bins - 0 and 1 - corresponding to their values. For the run count, we use six bins (from 0 to 5), where each bin is the run count (up to 4), and bin 5 contains run counts **more than 4**. For the fixation attributes - DT and FFD - we use the same binning scheme as described in [Klerke et al. \(2016\)](#). The binning scheme for fixation attributes is as follows:

- 0 if  $FV = 0$ ,
- 1 if  $FV > 0$  and  $FV \leq \mu - \sigma$ ,
- 2 if  $FV > \mu - \sigma$  and  $FV \leq \mu - 0.5 \times \sigma$ ,
- 3 if  $FV > \mu - 0.5 \times \sigma$  and  $FV \leq \mu + 0.5 \times \sigma$ ,
- 4 if  $FV > \mu + 0.5 \times \sigma$  and  $FV \leq \mu + \sigma$ ,
- 5 if  $FV > \mu + \sigma$ ,

where  $FV$  is the value of the given fixation attribute,  $\mu$  is the average fixation attribute value for

<sup>7</sup>The average time for the participants was about 2 hours, with the fastest completing the task in slightly under one and a half hours.

<sup>8</sup>As part of our data release, we will release the scores given by each annotator, as well as their justifications for their score

the reader and  $\sigma$  is the standard deviation.

## 5.6 Network Architecture

Figure 1 (b) shows the architecture of our proposed system, based on the co-attention based architecture described by [Zhang and Litman \(2018\)](#). Given an essay, we split the essay into sentences. For each sentence, we look-up the word embeddings for all words in the **Word Embedding** layer. The **4000 most frequent words** are used as the vocabulary, with all other words mapped to a special unknown token. This sequence of word embeddings is then sent through a Time-Delay Neural Network (TDNN), or **1-d Convolutional Neural Network** (CNN), of filter width  $k$ . The output from CNN is pooled using an attention layer - the **Word Level Attention Pooling Layer** - which results in a representation for every sentence. These sentence representations are then sent through a **Sentence Level LSTM Layer** and their output pooled in the **Sentence Level Attention Pooling Layer** to obtain the *sentence representation for the essay*.

A similar procedure is repeated for the source article. We then perform co-attention between the sentence representations of **the essay** and **the source article**. **Co-attention** is performed to learn similarities between the sentences in the essay and the source article. This is done as a way to ensure that the writer sticks to answering the prompt, rather than drifting off topic.

We now represent every sentence in the essay as a weighted combination of the sentence representation between the essay and the source article (Essay2Article). The weights are obtained from the output of the co-attention layer. The weights represent how each sentence in the essay are similar to the sentences in the source article. If a sentence in the essay has low weights this indicates that the sentence would be off topic. A similar procedure is repeated to get a weighted representation of sentences in the source article with respect to the essay (Article2Essay).

Finally, we send the sentence representation of the essay and article, through a dense layer (i.e. the **Modeling Layer**) to predict the final essay score, with a **sigmoid activation function**. As the essay scores are in the range [0, 1], we use sigmoid activation at the output layer. During prediction, we map the output scores from the sigmoid layer back to the original score range, minimizing the **mean squared error (MSE) loss**.

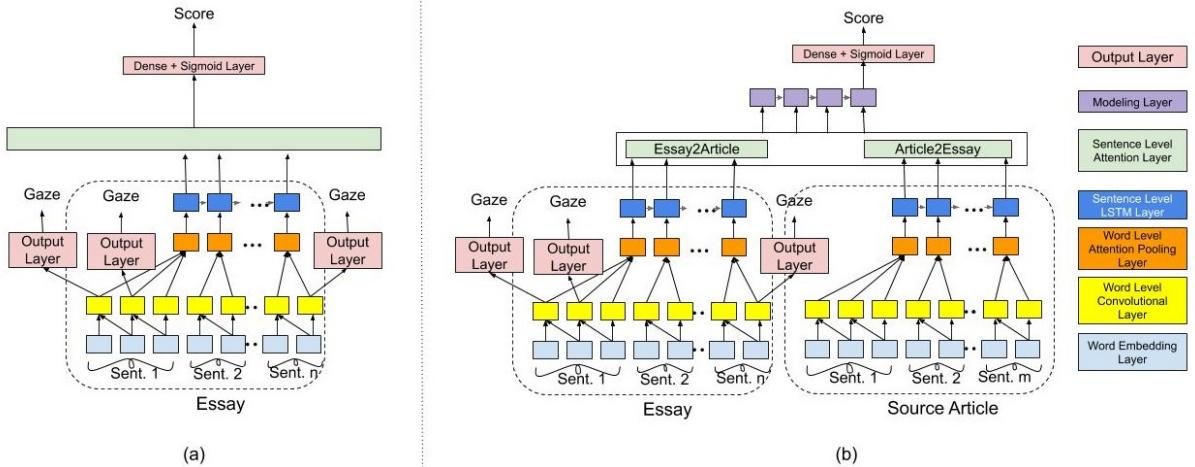


Figure 1: Architecture of the proposed gaze behaviour and essay scoring multi-task learning systems, namely (a) - the **Self-Attention** multi-task learning system, for an essay of  $n$  sentences - and (b) - the **Co-Attention** system for an essay of  $n$  sentences and a source article of  $m$  sentences.

For essay sets without a source article, we use the *Self-Attention* model proposed by Dong et al. (2017). This is a simpler model which does not consider the source article, and uses only the essay text. This is applicable whenever a source article is not present. Figure 1 (a) shows the architecture of the model. Like the earlier system, we get the *sentence representation of the essay* from the **Sentence Level LSTM Layer** and send it through the Dense Layer with a sigmoid activation function.

Gaze behaviour is learnt at the Word-Level Convolutional Layer in both the models because the gaze attributes are defined at the word-level, while the essay is scored at the document-level. The output from the CNN layer is sent through a linear layer followed by sigmoid activation for a particular gaze behaviour. For learning multiple gaze attributes simultaneously, we have multiple linear layers for each of the gaze attributes. In the multi-task setting, we also minimize the mean squared error of the learnt gaze behaviour and the actual gaze behaviour attribute value. We assign weights to each of the gaze behaviour loss functions to control the importance given to individual gaze behaviour learning tasks.

### 5.7 Network Hyperparameters

Table 3 gives the different hyperparameters which we used in our experiment. We use the 50 dimension GloVe pre-trained word embeddings (Pennington et al., 2014) trained on the Wikipedia 2014 + Gigawords 5 Corpus (6B tokens, 4K vocabulary, uncased). We run our experiments over a batch size of 100, for 100 epochs, and set the learning

Layer	Hyperparameter	Value
Embedding layer	Pre-trained embeddings	GloVe
	Embeddings dimensions	50
Word-level CNN	Kernel size	5
	Filters	100
Sentence-level LSTM	Hidden units	100
Network-wide	Batch size	100
	Epochs	100
	Learning rate	0.001
	Dropout rate	0.5
	Momentum	0.9

Table 3: Hyperparameters for our experiment.

rate as 0.001, and a dropout rate of 0.5. The Word-level CNN layer has a kernel size of 5, with 100 filters. The Sentence-level LSTM layer and modeling layer both have 100 hidden units. We use the RMSProp Optimizer (Dauphin et al., 2015) with a 0.001 initial learning rate and momentum of 0.9.

Gaze Feature	Gaze Feature Weight
Dwell Time	0.05
First Fixation Duration	0.05
IsRegression	0.01
Run Count	0.01
Skip	0.1

Table 4: This table shows the **best weights** assigned to the different gaze features from our grid search.

In addition to the network hyper-parameters, we also weigh the loss functions of the different gaze

behaviours differently, with weight levels of **0.5**, **0.1**, **0.05**, **0.01** and **0.001**. We use grid search and pick the weight giving the lowest mean-squared error on the *development* set. The best weights from grid search are **0.05** for DT and FFD, **0.01** for IR and RC, and **0.1** for Skip.

### 5.8 Experiment Configurations

To test our system on essay sets which we collected gaze behaviour, we run experiments using the following configurations. (a) **Self-Attention** - This is the implementation of Dong et al. (2017)'s system in Tensorflow by Zhang and Litman (2018). (b) **Co-Attention**. This is Zhang and Litman (2018)'s system<sup>9</sup>. (c) **Co-Attention+Gaze**. This is our system, which uses gaze behaviour.

In addition to this, we also run experiments on the **unseen** essay sets using the following *training* configurations. (a) **Only Prompt** - This uses our self-attention model, with the training data being only the essays from that essay set. We use this model, because there are no source articles for these essay sets. (b) **Extra Essays** - Here, we augment the training data of (a) with the **48 essays** for which we collect gaze behaviour data. (c) **Essays+Gaze** - Here, we augment the training data of (a) with the **48 essays** which we collect gaze behaviour data, and their corresponding *gaze data*. We also compare our results with a string kernel based system proposed by Cozma et al. (2018).

## 6 Results and Analysis

Table 5 reports the results of our experiments on the essay sets for which we collect the gaze behaviour data. The table is divided into 3 parts. The first part (*i.e.*, first 3 rows) are the reported results previously available deep-learning systems, namely Taghipour and Ng (2016), Dong and Zhang (2016), and Tay et al. (2018). The next 2 rows feature results using the self-attention (Dong et al., 2017) and co-attention (Zhang and Litman, 2018). The last row reports results using gaze behaviour on top of co-attention, *i.e.*, Co-Attention+Gaze. The first column is the different systems. The next 4 columns report the QWK results of each system for each of the 4 essay sets. The last column reports the Mean QWK value across all 4 essay sets.

Our system is able to outperform the Co-Attention system (Zhang and Litman, 2018) in all

the essay sets. Overall, it is also the best system - achieving the highest QWK results among all the systems in 3 out of the 4 essay sets (and the second-best in the other essay set). To test our hypothesis - that the model trained by learning gaze behaviour helps in automatic essay grading - we run the Paired T-Test. Our null hypothesis is: "Learning gaze behaviour to score an essay does not help any more than the self-attention and co-attention systems and whatever improvements we see are due to chance." We choose a significance level of  $p < 0.05$ , and observe that the improvements of our system are found to be statistically significant - rejecting the null hypothesis.

### 6.1 Results for Unseen Essay Sets

In order to run our experiments on **unseen** essay sets, we augment the training data with the gaze behaviour data collected. Since none of these essays have source articles, we use the self-attention model of Dong et al. (2017) as the baseline system. We now augment the gaze behaviour learning task as the auxiliary task and report the results in Table 6. The first column in the table is the different systems. The next 4 columns are the results for each of the unseen essay sets, and the last column is the mean QWK. From Table 6, we observe that our system which uses both the **extra 48 essays and their gaze behaviour** outperforms the other 2 configurations (**Only Prompt** and **Extra Essays**) across all 4 unseen essay sets. The improvement when learning gaze behaviour for **unseen** essay sets is statistically significant for  $p < 0.05$ .

### 6.2 Comparison with String Kernel System

Since Cozma et al. (2018) haven't released their data splits (train/test/dev), we ran their system with our data splits. We observed a mean QWK of **0.750** with the string kernel-based system on the essay sets where we have gaze behaviour data, and **0.685** on the unseen essay sets. One possible reason for this could be that while they used cross-validation, they may have used only a training-testing split (as compared to a train/test/dev split).

### 6.3 Analysis of Gaze Attributes

In order to see which of the gaze attributes are the most important, we ran ablation tests, where we ablate each gaze attribute. We found that the most important gaze behaviour attribute across all the essay sets is the Dwell Time, followed closely by the First Fixation Duration. One of the reasons

---

<sup>9</sup>The implementation of both systems can be downloaded from [here](#).

System	Prompt 3	Prompt 4	Prompt 5	Prompt 6	Mean QWK
Taghipour and Ng (2016)	0.683	0.795	<b>0.818</b>	0.813	0.777
Dong and Zhang (2016)	0.662	0.778	0.800	0.809	0.762
Tay et al. (2018)	0.695	0.788	0.815	0.810	0.777
Self-Attention (Dong et al., 2017)	0.677	0.807	0.806	0.809	0.775
Co-Attention (Zhang and Litman, 2018)	0.689†	0.809†	0.812†	0.813†	0.780†
Co-Attention+Gaze	<b>0.698*</b>	<b>0.818*</b>	0.815*	<b>0.821*</b>	<b>0.788*</b>

Table 5: Results of our experiments in scoring the essays (QWK values) from the essay sets where we collected gaze behaviour. The first 3 rows are results reported from other state-of-the-art deep learning systems. The next 2 rows are the results we obtained on existing systems - self-attention and co-attention - without gaze behaviour. The last row is the results from our system using gaze behaviour data (Co-Attention+Gaze). † denotes the baseline system performance, and \* denotes a statistically significant result of  $p < 0.05$  for the gaze behaviour system.

System	Prompt 1	Prompt 2	Prompt 7	Prompt 8	Mean QWK
Taghipour and Ng (2016)	0.775	<b>0.687</b>	0.805	0.594	0.715
Dong and Zhang (2016)	0.805	0.613	0.758	0.644	0.705
Tay et al. (2018)	0.832	0.684	0.800	0.697	0.753
Only Prompt (Dong et al. (2017))	0.816	0.667	0.792	0.678	0.738
Extra Essays	0.828†	0.672†	0.802†	0.685†	0.747†
Extra Essays + Gaze	<b>0.833</b>	0.681	<b>0.806*</b>	<b>0.699*</b>	<b>0.754*</b>

Table 6: Results of our experiments on the **unseen** essay sets our dataset. The first 3 rows are results reported from other state-of-the-art deep learning systems. The next 2 rows are the results obtained without using gaze behaviour (without and with the extra essays). The last row is the results from our system. † denotes the baseline system without gaze behaviour, and \* denotes a statistically significant result of  $p < 0.05$  for the gaze behaviour system.

Gaze Feature	Diff. in QWK
Dwell Time	<b>0.0137</b>
First Fixation Duration	0.0136
IsRegression	0.0090
Run Count	0.0110
Skip	0.0091

Table 7: Results of ablation tests for each gaze behaviour attribute across all the essay sets. The reported numbers are the **difference in QWK** before and after ablating the given gaze attribute. The number in **bold** denotes the best gaze attribute.

for this is the fact that both DT and FFD were very useful in detecting errors made by the essay writers. From Figure 2<sup>10</sup>, we observe that most of the longest dwell times have come at/around spelling mistakes (*tock* instead of *took*), or out-of-context words (*bay* instead of *by*), or incorrect phrases (*short cat*, instead of *short cut*). These errors force the reader to spend more time fixating on the word which we also mentioned earlier.

<sup>10</sup>We have given more examples in Appendix C.

The **normalized MSE** of each of the gaze features learnt by our system was between **0.125 to 0.128** for all the gaze behaviour attributes.

#### 6.4 Analysis Using Only a Native English Speaker

System	No	Native	All
Prompt 1	0.816	0.824	0.833
Prompt 2	0.667	0.679	0.681
Prompt 3	0.677	0.679	0.698
Prompt 4	0.807	0.812	0.818
Prompt 5	0.806	0.810	0.815
Prompt 6	0.809	0.815	0.821
Prompt 7	0.792	0.809	0.806
Prompt 8	0.678	0.679	0.699
<b>Mean QWK</b>	<b>0.757</b>	<b>0.764</b>	<b>0.771</b>

Table 8: Result using only gaze behaviour of the native speaker (Native), compared using no gaze behaviour (No) and gaze behaviour of all the readers (All).

We also ran our experiments using only the gaze behaviour of an annotator who was a **native** En-

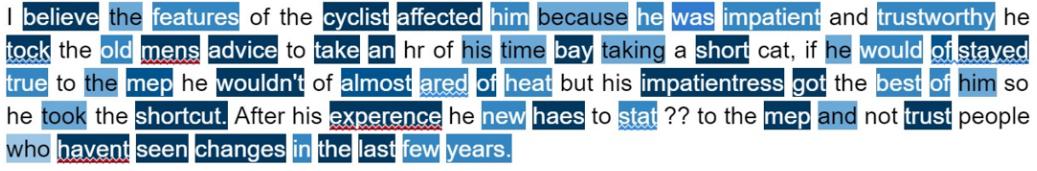


Figure 2: **Dwell Time** of one of the readers for one of the essays. The darker the background, the larger the bin.

glish speaker (as opposed to the rest of our annotators who were just *fluent* English speakers). Table 8 shows the results of those experiments. We observed a mean QWK of **0.779** for the seen essay sets, and a mean QWK of **0.748** for the essays sets where we have no gaze data. The difference in performance between both our systems (i.e. with only native speaker and with all annotators) were found to be statistically significant with  $p = 0.0245^{11}$ . Similarly, the improvement in performance using the native English speaker, compared to not using any gaze behaviour was also found to be statistically significant for  $p = 0.0084$ .

## 7 Conclusion and Future Work

In this paper, we describe how learning gaze behaviour can help AEG in a multi-task learning setup. We explained how we created a resource by collecting gaze behaviour data, and using multi-task learning we are able to achieve better results over a state-of-the-art system developed by [Zhang and Litman \(2018\)](#) for the essay sets which we collected gaze behaviour data from. We also analyze the transferability of gaze behaviour patterns across essay sets by training a multi-task learning model on **unseen** essay sets (i.e. essay sets where we have no gaze behaviour data), thereby establishing that learning gaze behaviour improves automatic essay grading.

In the future, we would like to look at using gaze behaviour to help in cross-domain AEG. This is done mainly when we don't have enough training examples in our essay set. We would also like to explore the possibility of generating textual feedback (rather than just a number, denoting the score of the essay) based on the justifications that the annotators gave for their grades.

<sup>11</sup>The p-values for the different experiments are in Appendix D.

## References

- Maria Barrett, Joachim Bingel, Nora Hollenstein, Marek Rei, and Anders Søgaard. 2018. [Sequence classification with human attention](#). In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 302–312, Brussels, Belgium. Association for Computational Linguistics.
- Maria Barrett, Joachim Bingel, Frank Keller, and Anders Søgaard. 2016. [Weakly supervised part-of-speech tagging using eye-tracking data](#). In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 579–584, Berlin, Germany. Association for Computational Linguistics.
- Rich Caruana. 1998. [Multitask Learning](#), pages 95–133. Springer US, Boston, MA.
- Jacob Cohen. 1968. Weighted kappa: Nominal scale agreement provision for scaled disagreement or partial credit. *Psychological bulletin*, 70(4):213.
- Mădălina Cozma, Andrei Butnaru, and Radu Tudor Ionescu. 2018. [Automated essay scoring with string kernels and word embeddings](#). In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 503–509, Melbourne, Australia. Association for Computational Linguistics.
- Yann Dauphin, Harm De Vries, and Yoshua Bengio. 2015. Equilibrated adaptive learning rates for non-convex optimization. In *Advances in neural information processing systems*, pages 1504–1512.
- Fei Dong and Yue Zhang. 2016. [Automatic features for essay scoring – an empirical study](#). In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1072–1077, Austin, Texas. Association for Computational Linguistics.
- Fei Dong, Yue Zhang, and Jie Yang. 2017. [Attention-based recurrent convolutional neural network for automatic essay scoring](#). In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 153–162, Vancouver, Canada. Association for Computational Linguistics.
- Ana V González-Garduño and Anders Søgaard. 2018. Learning to predict readability using eye-movement data from natives and learners. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

- Ana Valeria González-Garduño and Anders Søgaard. 2017. [Using gaze to predict text readability](#). In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 438–443, Copenhagen, Denmark. Association for Computational Linguistics.
- John Hale, Chris Dyer, Adhiguna Kuncoro, and Jonathan Brennan. 2018. [Finding syntax in human encephalography with beam search](#). In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2727–2736, Melbourne, Australia. Association for Computational Linguistics.
- Nora Hollenstein and Ce Zhang. 2019. [Entity recognition at first sight: Improving NER with eye movement information](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1–10, Minneapolis, Minnesota. Association for Computational Linguistics.
- Kenneth Holmqvist, Marcus Nyström, Richard Andersson, Richard Dewhurst, Halszka Jarodzka, and Joost Van de Weijer. 2011. *Eye tracking: A comprehensive guide to methods and measures*. OUP Oxford.
- Marcel A Just and Patricia A Carpenter. 1980. A theory of reading: From eye fixations to comprehension. *Psychological review*, 87(4):329.
- Alan Kennedy, Robin Hill, and Joël Pynte. 2003. The dundee corpus. In *Proceedings of the 12th European conference on eye movement*.
- Sigríður Klerke, Yoav Goldberg, and Anders Søgaard. 2016. [Improving sentence compression by learning to predict gaze](#). In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1528–1533, San Diego, California. Association for Computational Linguistics.
- Yunfei Long, Rong Xiang, Qin Lu, Chu-Ren Huang, and Minglei Li. 2019. Improving attention model based on cognition grounded data for sentiment analysis. *IEEE Transactions on Affective Computing*.
- Sandeep Mathias, Diptesh Kanodia, Kevin Patel, Samarth Agrawal, Abhijit Mishra, and Pushpak Bhattacharyya. 2018. [Eyes are the windows to the soul: Predicting the rating of text quality using gaze behaviour](#). In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2352–2362, Melbourne, Australia. Association for Computational Linguistics.
- Abhijit Mishra and Pushpak Bhattacharyya. 2018. *Cognitively Inspired Natural Language Processing: An Investigation Based on Eye-tracking*. Springer.
- Abhijit Mishra, Diptesh Kanodia, and Pushpak Bhattacharyya. 2016. [Predicting readers' sarcasm understandability by modeling gaze behavior](#).
- Abhijit Mishra, Diptesh Kanodia, Seema Nagar, Kuntal Dey, and Pushpak Bhattacharyya. 2017. [Scanpath complexity: Modeling reading effort using gaze information](#).
- Abhijit Mishra, Srikanth Tamilvelvam, Riddhiman Dasgupta, Seema Nagar, and Kuntal Dey. 2018. Cognition-cognizant sentiment analysis with multi-task subjectivity summarization based on annotators' gaze behavior. In *Thirty-Second AAAI Conference on Artificial Intelligence*.
- Ellis B Page. 1966. The imminence of... grading essays by computer. *The Phi Delta Kappan*, 47(5):238–243.
- Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. [Glove: Global vectors for word representation](#). In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.
- Mark D Shermis and Jill Burstein. 2013. *Handbook of automated essay evaluation: Current applications and new directions*. Routledge.
- Abhinav Deep Singh, Poojan Mehta, Samar Husain, and Rajkumar Rajakrishnan. 2016. [Quantifying sentence complexity based on eye-tracking measures](#). In *Proceedings of the Workshop on Computational Linguistics for Linguistic Complexity (CL4LC)*, pages 202–212, Osaka, Japan. The COLING 2016 Organizing Committee.
- Kaveh Taghipour and Hwee Tou Ng. 2016. [A neural approach to automated essay scoring](#). In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1882–1891, Austin, Texas. Association for Computational Linguistics.
- Yi Tay, Minh Phan, Luu Anh Tuan, and Siu Cheung Hui. 2018. [Skipflow: Incorporating neural coherence features for end-to-end automatic text scoring](#).
- Haoran Zhang and Diane Litman. 2018. [Co-attention based neural network for source-dependent essay scoring](#). In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 399–409, New Orleans, Louisiana. Association for Computational Linguistics.

## A Source Article (Prompt 6)

*The Mooring Mast*, by Marcia Amidon Lusted

When the Empire State Building was conceived, it was planned as the world's tallest building, taller even than the new Chrysler Building that was being constructed at Forty-second Street and Lexington

Avenue in New York. At seventy-seven stories, it was the tallest building before the Empire State began construction, and Al Smith was determined to outstrip it in height.

The architect building the Chrysler Building, however, had a trick up his sleeve. He secretly constructed a 185-foot spire inside the building, and then shocked the public and the media by hoisting it up to the top of the Chrysler Building, bringing it to a height of 1,046 feet, 46 feet taller than the originally announced height of the Empire State Building.

Al Smith realized that he was close to losing the title of world's tallest building, and on December 11, 1929, he announced that the Empire State would now reach the height of 1,250 feet. He would add a top or a hat to the building that would be even more distinctive than any other building in the city. John Tauranac describes the plan:

"[The top of the Empire State Building] would be more than ornamental, more than a spire or dome or a pyramid put there to add a desired few feet to the height of the building or to mask something as mundane as a water tank. Their top, they said, would serve a higher calling. The Empire State Building would be equipped for an age of transportation that was then only the dream of aviation pioneers."

This dream of the aviation pioneers was travel by dirigible, or zeppelin, and the Empire State Building was going to have a mooring mast at its top for docking these new airships, which would accommodate passengers on already existing transatlantic routes and new routes that were yet to come.

### A.1 The Age of Dirigibles

By the 1920s, dirigibles were being hailed as the transportation of the future. Also known today as blimps, dirigibles were actually enormous steel-framed balloons, with envelopes of cotton fabric filled with hydrogen and helium to make them lighter than air. Unlike a balloon, a dirigible could be maneuvered by the use of propellers and rudders, and passengers could ride in the gondola, or enclosed compartment, under the balloon.

Dirigibles had a top speed of eighty miles per hour, and they could cruise at seventy miles per hour for thousands of miles without needing refueling. Some were as long as one thousand feet, the same length as four blocks in New York City. The one obstacle to their expanded use in New

York City was the lack of a suitable landing area. Al Smith saw an opportunity for his Empire State Building: A mooring mast added to the top of the building would allow dirigibles to anchor there for several hours for refueling or service, and to let passengers off and on. Dirigibles were docked by means of an electric winch, which hauled in a line from the front of the ship and then tied it to a mast. The body of the dirigible could swing in the breeze, and yet passengers could safely get on and off the dirigible by walking down a gangplank to an open observation platform.

The architects and engineers of the Empire State Building consulted with experts, taking tours of the equipment and mooring operations at the U.S. Naval Air Station in Lakehurst, New Jersey. The navy was the leader in the research and development of dirigibles in the United States. The navy even offered its dirigible, the Los Angeles, to be used in testing the mast. The architects also met with the president of a recently formed airship transport company that planned to offer dirigible service across the Pacific Ocean.

When asked about the mooring mast, Al Smith commented:

"[It's] on the level, all right. No kidding. We're working on the thing now. One set of engineers here in New York is trying to dope out a practical, workable arrangement and the Government people in Washington are figuring on some safe way of mooring airships to this mast."

### A.2 Designing the Mast

The architects could not simply drop a mooring mast on top of the Empire State Building's flat roof. A thousand-foot dirigible moored at the top of the building, held by a single cable tether, would add stress to the building's frame. The stress of the dirigible's load and the wind pressure would have to be transmitted all the way to the building's foundation, which was nearly eleven hundred feet below. The steel frame of the Empire State Building would have to be modified and strengthened to accommodate this new situation. Over sixty thousand dollars' worth of modifications had to be made to the building's framework.

Rather than building a utilitarian mast without any ornamentation, the architects designed a shiny glass and chrome-nickel stainless steel tower that would be illuminated from inside, with a stepped-back design that imitated the overall shape of the

building itself. The rocket-shaped mast would have four wings at its corners, of shiny aluminum, and would rise to a conical roof that would house the mooring arm. The winches and control machinery for the dirigible mooring would be housed in the base of the shaft itself, which also housed elevators and stairs to bring passengers down to the eighty-sixth floor, where baggage and ticket areas would be located.

The building would now be 102 floors, with a glassed-in observation area on the 101st floor and an open observation platform on the 102nd floor. This observation area was to double as the boarding area for dirigible passengers.

Once the architects had designed the mooring mast and made changes to the existing plans for the building's skeleton, construction proceeded as planned. When the building had been framed to the 85th floor, the roof had to be completed before the framing for the mooring mast could take place. The mast also had a skeleton of steel and was clad in stainless steel with glass windows. Two months after the workers celebrated framing the entire building, they were back to raise an American flag again—this time at the top of the frame for the mooring mast.

### A.3 The Fate of the Mast

The mooring mast of the Empire State Building was destined to never fulfill its purpose, for reasons that should have been apparent before it was ever constructed. The greatest reason was one of safety: Most dirigibles from outside of the United States used hydrogen rather than helium, and hydrogen is highly flammable. When the German dirigible Hindenburg was destroyed by fire in Lakehurst, New Jersey, on May 6, 1937, the owners of the Empire State Building realized how much worse that accident could have been if it had taken place above a densely populated area such as downtown New York.

The greatest obstacle to the successful use of the mooring mast was nature itself. The winds on top of the building were constantly shifting due to violent air currents. Even if the dirigible were tethered to the mooring mast, the back of the ship would swivel around and around the mooring mast. Dirigibles moored in open landing fields could be weighted down in the back with lead weights, but using these at the Empire State Building, where they would be dangling high above pedestrians on

the street, was neither practical nor safe.

The other practical reason why dirigibles could not moor at the Empire State Building was an existing law against airships flying too low over urban areas. This law would make it illegal for a ship to ever tie up to the building or even approach the area, although two dirigibles did attempt to reach the building before the entire idea was dropped. In December 1930, the U.S. Navy dirigible Los Angeles approached the mooring mast but could not get close enough to tie up because of forceful winds. Fearing that the wind would blow the dirigible onto the sharp spires of other buildings in the area, which would puncture the dirigible's shell, the captain could not even take his hands off the control levers.

Two weeks later, another dirigible, the Goodyear blimp Columbia, attempted a publicity stunt where it would tie up and deliver a bundle of newspapers to the Empire State Building. Because the complete dirigible mooring equipment had never been installed, a worker atop the mooring mast would have to catch the bundle of papers on a rope dangling from the blimp. The papers were delivered in this fashion, but after this stunt the idea of using the mooring mast was shelved. In February 1931, Irving Clavan of the building's architectural office said, "The as yet unsolved problems of mooring air ships to a fixed mast at such a height made it desirable to postpone to a later date the final installation of the landing gear."

By the late 1930s, the idea of using the mooring mast for dirigibles and their passengers had quietly disappeared. Dirigibles, instead of becoming the transportation of the future, had given way to airplanes. The rooms in the Empire State Building that had been set aside for the ticketing and baggage of dirigible passengers were made over into the world's highest soda fountain and tea garden for use by the sightseers who flocked to the observation decks. The highest open observation deck, intended for disembarking passengers, has never been open to the public.

## B Annotator Profiles

Table 9 summarizes the profiles of the different annotators. It details each of the 8 annotators, their sex, age, occupations, L1 / native languages, their performance in a high school Examination in English and whether or not they have had experience as a TA. The last 3 columns are their performance

ID	Sex	Age	Occupation	TA?	L1 Language	English Score	QWK	Correct	Close
Annotator 1	Male	23	Masters student	Yes	Hindi	94%	0.611	19	41
Annotator 2	Male	18	Undergraduate	Yes	Marathi	95%	0.587	24	41
Annotator 3	Male	31	Research scholar	Yes	Marathi	85%	0.659	21	43
Annotator 4	Male	28	Software engineer	Yes	English	96%	0.659	26	44
Annotator 5	Male	30	Research scholar	Yes	Gujarati	92%	0.600	19	42
Annotator 6	Female	22	Masters student	Yes	Marathi	95%	0.548	19	40
Annotator 7	Male	19	Undergraduate	Yes	Marathi	93%	0.732	21	46
Annotator 8	Male	28	Masters student	Yes	Gujarati	94%	0.768	29	45

Table 9: Profile of the annotators

on the annotation grading task, where QWK is their agreement with the ground truth scores, Correct is the number of times (out of 48) where their essay scores matched with the ground truth scores, and Close is the number of times (out of 48) where they disagreed with the ground truth score by at most 1 grade point.

## C Heat Map Examples

### C.1 Different Gaze Features

Here, we show examples of heat maps for different gaze behaviour attributes of one of our readers.

1. Figure 3 shows the dwell time of the reader.
2. Figure 4 shows the heat map of the first fixation duration of a reader.
3. Figure 5 shows the heat map of the IsRegression feature - i.e. whether or not the reader regressed from a particular word.
4. Figure 6 shows the heat map of the Run Count of the reader.
5. Figure 7 shows the words that the reader read (highlighted) and skipped (unhighlighted).

### C.2 Dwell Times of Good and Bad Essays

Figures 8 and 9 show the dwell time heat maps of a reader as he reads a good essay and a bad essay respectively. For the bad essay, notice the amount of a lot more darker blues compared to the good essay.

## D P-Values

In this section, we report the p-values and other results for our experiments.

### D.1 Source-Dependent Essay Set's p-values

The results shown here in Table 10 are the p-values for the different essay sets with and without gaze from Table 5.

Essay Set	p-value
Prompt 3	0.0042
Prompt 4	0.0109
Prompt 5	0.0133
Prompt 6	0.0003

Table 10: Source-Dependent essay set's p-values

### D.2 Unseen Essay Set's p-values

The results shown here in Table 10 are the p-values for the different essay sets with and without gaze from Table 6.

Essay Set	p-value
Prompt 1	0.0887
Prompt 2	0.1380
Prompt 7	0.0393
Prompt 8	0.0315

Table 11: Unseen Essay's p-values

### D.3 Native Gaze vs. No Gaze & All Gaze p-values

The results shown in Table 12 are the p-values for the essay sets using the gaze behaviour of a native English speaker compared to not using gaze behaviour, and using gaze behaviour of all readers.

Essay Set	No vs. Native	Native vs. All
Prompt 1	0.1407	0.0471
Prompt 2	0.0161	0.9161
Prompt 3	0.3239	0.0239
Prompt 4	0.0810	0.0805
Prompt 5	0.4971	0.4010
Prompt 6	0.2462	0.2961
Prompt 7	0.0189	0.0098
Prompt 8	0.8768	0.0068

Table 12: No gaze vs. native gaze and native gaze vs. all gaze p-values.

I believe the features of the cyclist affected him because he was impatient and trustworthy he took the old mens advice to take an hr of his time bay taking a short cat, if he would of stayed true to the mep he wouldn't of almost ared of heat but his impatientress got the best of him so he took the shortcut. After his experence he new haes to stat ?? to the mep and not trust people who havent seen changes in the last few years.

Figure 3: Sample heat map of the dwell of a reader for the text. The darker the blue, the larger the bin, and the longer the dwell time.

I believe the features of the cyclist affected him because he was impatient and trustworthy he took the old mens advice to take an hr of his time bay taking a short cat, if he would of stayed true to the mep he wouldn't of almost ared of heat but his impatientress got the best of him so he took the shortcut. After his experence he new haes to stat ?? to the mep and not trust people who havent seen changes in the last few years.

Figure 4: Sample heat map of the first fixation duration of a reader for the text. The darker the blue, the larger the bin, and the longer the first fixation duration.

I believe the features of the cyclist affected him because he was impatient and trustworthy he took the old mens advice to take an hr of his time bay taking a short cat, if he would of stayed true to the mep he wouldn't of almost ared of heat but his impatientress got the best of him so he took the shortcut. After his experence he new haes to stat ?? to the mep and not trust people who havent seen changes in the last few years.

Figure 5: Sample heat map of the Is Regression feature of a reader for the text. The highlighted words denote words that the reader regressed from.

I believe the features of the cyclist affected him because he was impatient and trustworthy he took the old mens advice to take an hr of his time bay taking a short cat, if he would of stayed true to the mep he wouldn't of almost ared of heat but his impatientress got the best of him so he took the shortcut. After his experence he new haes to stat ?? to the mep and not trust people who havent seen changes in the last few years.

Figure 6: Sample heat map of the run count of a reader for the text. The darker the blue, the larger the bin, and the higher the run count.

I believe the features of the cyclist affected him because he was impatient and trustworthy he took the old mens advice to take an hr of his time bay taking a short cat, if he would of stayed true to the mep he wouldn't of almost ared of heat but his impatientress got the best of him so he took the shortcut. After his experence he new haes to stat ?? to the mep and not trust people who havent seen changes in the last few years.

Figure 7: Sample heat map of the Skip feature of a reader for the text. The unhighlighted words denote words that the reader skipped.

The engineers involved in the creation of the Empire State Building were forced to confront reality when an array of obstacles presented themselves during the time in which they were trying to dock dirigibles. The primary problem was the usefulness of this dock creation. Though this idea was innovative, it was not practical, as dirigibles were never destined to be a popular source of transport. The malfunction in the creation of the idea was its focus. This is because the goal in this work was not to create a successful dock, but to add footage to the building. If the focus had been different, the outcome may have been more rewarding. Technical problems also arose. Based on laws, safety, and practicality it could not function. Most dirigibles from outside the United States used hydrogen, creating an extreme fire hazard in a highly populated place that would transform into a deathtrap. The anchor for the blimp would only secure it at one point allowing the blimp to spin around, dangerously in the wind. Lead weights, the only solution to this, would disrupt pedestrians. There was also an existing law against airships flying too low over urban areas, making the project completely unpractical. Both attempts at reaching the building failed, displaying the reality of the flaws. Winds and other complications were preventative. All in all, the builders were destined to be unsuccessful with the plethora of flaws in this project.

Figure 8: Dwell Time for a reader for an essay which he scored well.

Immigrants from Cuba to the USA usually had to undergo a tough transition. It took buckets full of courage and moderation to get through this huge transition. Narciso Rodriguez's family went through this move using family power and the love they had for each other. It was a big change for them to be in a one room apartment to a three room apartment. Through the memoir there was a mood of satisfaction and love for Cuba. For example, the memoir stated many times how much Narciso enjoyed the Cuban food and music and traditions. Also the satisfaction mood comes out when Narciso talks about how fortunate he is that his parents were willing to take the risk of moving to New Jersey to give him a better life. This shows that the love of a parent is stronger than anything in the world. Another mood is relieve, for example Narciso is relieved that his family made it to New Jersey safely and sound. This article is a perfect example that family is all you have in the end and that no matter what they are there for you.

Figure 9: Dwell Time for a reader for an essay which he scored badly.

# Multi-Source Attention for Unsupervised Domain Adaptation

Xia Cui

University of Liverpool

Xia.Cui@liverpool.ac.uk

Danushka Bollegala\*

University of Liverpool, Amazon

danushka@liverpool.ac.uk

## Abstract

We model source-selection in multi-source Unsupervised Domain Adaptation (UDA) as an attention-learning problem, where we learn attention over the sources per given target instance. We first independently learn source-specific classification models, and a relatedness map between sources and target domains using pseudo-labelled target domain instances. Next, we learn domain-attention scores over the sources for aggregating the predictions of the source-specific models. Experimental results on two cross-domain sentiment classification datasets show that the proposed method reports consistently good performance across domains, and at times outperforming more complex prior proposals. Moreover, the computed domain-attention scores enable us to find explanations for the predictions made by the proposed method.<sup>1</sup>

## 1 Introduction

Domain adaptation (DA) considers the problem of generalising a model learnt using the data from a particular source domain to a different target domain (Zhang et al., 2015). Although most DA methods consider adapting to a target domain from a single source domain (Blitzer et al., 2006, 2007; Ganin et al., 2016), often it is difficult to find a suitable single source to adapt from, and one must consider multiple sources. For example, in sentiment classification, each product category is considered as a *domain* (Blitzer et al., 2006), resulting in a multi-domain adaptation setting.

Unsupervised DA (UDA) is a special case of DA where labelled instances are not available for

Danushka Bollegala holds concurrent appointments as a Professor at University of Liverpool and as an Amazon Scholar. This paper describes work performed at the University of Liverpool and is not associated with Amazon.

<sup>1</sup>Source code available at <https://github.com/LivNLP/multi-source-attention>

the target domain. Existing approaches for UDA can be categorised into pivot-based and instance-based methods. Pivots refer to the features common to both source and target domains (Blitzer et al., 2006). Pivot-based single-source domain adaptation methods, such as Structural Correspondence Learning (SCL; Blitzer et al., 2006, 2007) and Spectral Feature Alignment (SFA; Pan et al., 2010), first select a set of pivots and then project the source and target domain documents into a shared space. Next, a prediction model is learnt in this shared space. However, these methods fail in multi-source settings because it is challenging to find pivots across all sources such that a single shared projection can be learnt. Similarly, instance-based methods, such as Stacked Denoising Autoencoders (SDA; Glorot et al., 2011) and marginalised SDA (mSDA; Chen et al., 2012) minimise the loss between the original inputs and their reconstructions. Not all of the source domains are appropriate for learning transferable projections for a particular target domain. Adapting from an unrelated source can result in poor performance on the given target, which is known as *negative transfer* (Rosenstein et al., 2005; Pan and Yang, 2010; Guo et al., 2018).

Prior proposals for multi-source UDA can be broadly classified into methods that: (a) first select a source domain and then select instances from that source domain to adapt to a given target domain test instance (Ganin et al., 2016; Kim et al., 2017; Zhao et al., 2018; Guo et al., 2018); (b) pool all source domain instances together and from this pool select instances to adapt to a given target domain test instance (Chattopadhyay et al., 2012); (c) pick a source domain and use all instances in that source (source domain selection) (Schultz et al., 2018); and (d) pick all source domains and use all instances (utilising all instances) (Aue and Gamon, 2005; Bollegala et al., 2011; Wu and Huang, 2016).

In contrast, we propose a multi-source UDA

method that systematically addresses the various challenges in multi-source UDA.

- Although in UDA we have labelled instances in each source domain, its number is significantly smaller than that of the unlabelled instances in the same domain. For example, in the Amazon product review dataset released by Blitzer et al. (2007) there are 73679 unlabelled instances in total across the four domains, whereas there are only 4800 labelled instances. To increase the labelled instances in a source domain, we induce pseudo-labels for the unlabelled instances in each source domain using self-training as in § 3.1.
- In UDA, we have no labelled data for the target domain. To address this challenge, we infer pseudo-labels for the target domain’s unlabelled training instances by majority voting over the classifiers trained from each source domain, using both labelled and pseudo-labelled instances as in § 3.1.
- Given that the pseudo-labels inferred for the target domain instances are inherently more noisier compared to the manually labelled source domain instances, we propose a method to identify a subset of prototypical target domain instances for DA using document embedding similarities as described in § 3.2.
- The accuracy of UDA is upper-bounded by the  $\mathcal{H}$ -divergence between a source and a target domain (Kifer et al., 2004; Ben-David et al., 2006, 2009). Therefore, when predicting the label of a target domain test instance, we must select only the relevant labelled instances from a source domain. We propose a method to learn such a *relatedness map* between source and target domains in § 3.3.
- To reduce negative transfer, for each target domain test instance we dynamically compute a *domain-attention* score that expresses the relevance of a source domain. For this purpose, we represent each domain by a *domain embedding*, which we learn in an end-to-end fashion using the target domain’s pseudo-labelled instances as detailed in § 3.4.

We evaluate the proposed method on two standard cross-domain sentiment classification benchmarks for UDA. We find that both pseudo-labels and domain-attention scores contribute toward improving the classification accuracy for a target domain. The proposed method reports consistently good

performance in both datasets and across multiple domains. Although the proposed method does not outperform more complex UDA methods in some cases, using the domain-attention scores, we are able to retrieve justifications for the predicted labels.

## 2 Related Work

In § 1 we already mentioned prior proposals for single-source DA and this section discusses multi-source DA, which is the main focus of this paper. Bollegala et al. (2011) created a sentiment sensitive thesaurus (SST) using the data from the union of multiple source domains to train a cross-domain sentiment classifier. The SST is used to expand feature spaces during train and test times. The performance of SST depends heavily on the selection of pivots (Cui et al., 2017; Li et al., 2017). Wu and Huang (2016) proposed a sentiment DA method from multiple sources (SDAMS) by introducing two components: a sentiment graph and a domain similarity measure. The sentiment graph is extracted from unlabelled data. Similar to SST, SDAMS uses data from multiple sources to maximise the available labelled data. Guo et al. (2020) proposed a mixture of distance measures and used a multi-arm bandit to dynamically select a single source during training. However, in our proposed method all domains are selected and contributing differently as specified by their domain-attention weights for each train and test instance. Moreover, we use only one distance measure and is easier to implement.

Recently, Adversarial NNs have become popular in DA (Ganin et al., 2016; Zhao et al., 2018; Guo et al., 2018). Adversarial training is used to reduce the discrepancy between source and target domains (Ding et al., 2019). Domain-Adversarial Neural Networks (DANN; Ganin et al., 2016) use a gradient reversal layer to learn domain independent features for a given task. Multiple Source Domain Adaptation with Adversarial Learning (MDAN; Zhao et al., 2018) generalises DANN and aims to learn domain independent features while being relevant to the target task. Li et al. (2017) proposed End-to-End Adversarial Memory Network (AMN), inspired by memory networks (Sukhbaatar et al., 2015), and automatically capture pivots using an attention mechanism. Guo et al. (2018) proposed an UDA method using a mixture of experts for each domain. They model the domain relations

using a *point-to-set* distance metric to the encoded training matrix for source domains. Next, they perform joint training over all domain-pairs to update the parameters in the model by *meta-training*. However, they ignore the available unlabelled instances for the source domain. Adversarial training methods have shown to be sensitive to the hyper parameter values and require problem-specific techniques (Mukherjee et al., 2018). Kim et al. (2017) modeled domain relations using *example-to-domain* based on an attention mechanism. However, the attention weights are learnt using source domain training data in a supervised manner. Following a self-training approach, Chattpadhyay et al. (2012) proposed a two-stage weighting framework for multi-source DA that first computes the weights for features from different source domains using Maximum Mean Discrepancy (MMD; Borgwardt et al., 2006). Next, they generate pseudo labels for the target unlabelled instances using a classifier learnt from the multiple source domains. Finally, a classifier is trained on the pseudo-labelled instances for the target domain. Their method requires labelled data for the target domain, which is a *supervised* DA setting, different from the UDA setting we consider in this paper. Our proposed method uses self-training to assign pseudo-labels for the unlabelled target instances, and learn an embedding for each domain using an attention mechanism.

### 3 Multi-Source Domain Attention

Let us assume that we are given  $N$  source domains,  $S_1, S_2, \dots, S_N$ , and required to adapt to a target domain  $T$ . Moreover, let us denote the labelled instances in  $S_i$  by  $\mathcal{S}_i^L$  and unlabelled instances by  $\mathcal{S}_i^U$ . For  $T$  we have only unlabelled instances  $\mathcal{T}^U$  in UDA. Our goal is to learn a binary classifier<sup>2</sup> to predict labels ( $\in \{0, 1\}$ ) for the target domain instances using  $\mathcal{S}^L = \cup_{i=1}^N \mathcal{S}_i^L$ ,  $\mathcal{S}^U = \cup_{i=1}^N \mathcal{S}_i^U$  and  $\mathcal{T}^U$ . We denote labelled and unlabelled instances in  $S_i$  by respectively  $x_i^L$  and  $x_i^U$ , whereas instances in  $T$  are denoted by  $x_T$ . To simplify the notation, we drop the superscripts  $L$  and  $U$  when it is clear from the context whether the instance is respectively labelled or not.

The steps of our proposed method can be summarised as follows: (a) use labelled and unlabelled

<sup>2</sup>Although we consider binary sentiment classification as an evaluation task in this paper, the proposed method can be easily extended to multi-class classification settings by making 1-vs-rest prediction tasks (Rifkin and Klautau, 2004).

instances from each of the source domains to learn classifiers that can predict the label for a given instance. Next, develop a majority voter and use it to predict the *pseudo-labels* for the target domain unlabelled instances  $\mathcal{T}^U$  (§ 3.1); (b) compute a *relatedness map* between the target domain’s pseudo-labelled instances,  $\mathcal{T}^{L*}$ , and source domains’ labelled instances  $\mathcal{S}^L$  (§ 3.3); (c) compute *domain-attention* weights for each source domain (§ 3.4); (d) jointly learn a model based on the relatedness map and the domain-attention weights for predicting labels for the target domain’s test instances (§ 3.5).

#### 3.1 Pseudo-Label Generation

In UDA, we have only unlabelled data for the target domain. Therefore, we first infer pseudo-labels for the target domain instances  $\mathcal{T}^U$  by self-training (Abney, 2007) following Algorithm 1. Specifically, we first train a predictor  $f_i$  for the  $i$ -th source domain using only its labelled instances  $\mathcal{S}_i^L$  using a base learner  $\Gamma$  (Line 1-2). Any classification algorithm that can learn a predictor  $f_i$  that can compute the probability,  $f_i(x, y)$ , of a given instance  $x$  belonging to the class  $y$  can be used as  $\Gamma$ . In our experiments, we use logistic regression for its simplicity and popularity in prior UDA work (Bollegrala et al., 2011; Bollegala et al., 2013).

Next, for each unlabelled instance in the selected source domain, we compute the probability of it belonging to each class and find the most probable class label. If the probability of the most likely class is greater than the given confidence threshold  $\tau \in [0, 1]$ , we will append that instance to the current labelled training set. This enables us to increase the labelled instances for the source domains, which is important for learning accurate classifiers when the amount of labelled instances available is small. After processing all unlabelled instances in  $S_i$ , we train the final classifier  $f_i$  for  $S_i$  using both original and pseudo-labelled instances. Finally, we predict a pseudo-label for a target domain instance as the majority vote,  $f^* \in \{0, 1\}$ , over the predictions made by the individual classifiers  $f_i$ .

#### 3.2 Prototype Selection

Selecting the highest confident pseudo-labelled instances for training a classifier for the target domain as done in prior work (Zhou and Li, 2005; Abney, 2007; Søgaard, 2010; Ruder and Plank, 2018) does not guarantee that those instances will be the most

---

**Algorithm 1** Multi-Source Self-Training

---

**Input:** source domains’ labelled instances  $\mathcal{S}_1^L, \dots, \mathcal{S}_N^L$ , source domains’ unlabelled instances  $\mathcal{S}_1^U, \dots, \mathcal{S}_N^U$  and target domain’s unlabelled instances  $\mathcal{T}^U$ , target classes  $\mathcal{Y}$ , base learner  $\Gamma$  and the classification confidence threshold  $\tau$ .  
**Output:** multi-source self-training classifier  $f^*$

```

1: for  $i = 1$  to  $N$  do
2:    $\mathcal{L}_i \leftarrow \mathcal{S}_i^L$ 
3:    $f_i \leftarrow \Gamma(\mathcal{L}_i)$ 
4:   for  $x \in \mathcal{S}_i^U$  do
5:      $\hat{y} = \arg \max_{y \in \mathcal{Y}} f_i(x, y)$ 
6:     if  $f_i(x, \hat{y}) > \tau$  then
7:        $\mathcal{L}_i \leftarrow \mathcal{L}_i \cup \{(x, \hat{y})\}$ 
8:     end if
9:   end for
10:   $f_i \leftarrow \Gamma(\mathcal{L}_i)$ 
11: end for
12: return majority voter  $f^*$  over  $f_1, \dots, f_N$ .
```

---

suitable ones for adapting to the target domain, which was not considered during the self-training stage. For example, some target instances might not be good prototypical examples of the target domain and we would not want to use the pseudo-labels induced for those instances when training a classifier for the target domain. To identify instances in the target domain that are better prototypes, we first encode each target instance by a vector and select the instances that are closest to the centroid,  $\mathbf{c}_T$ , of the target domain instances given by (1).

$$\mathbf{c}_T = \frac{1}{|\mathcal{T}^U|} \sum_{x \in \mathcal{T}^U} x \quad (1)$$

In the case of text documents  $x$ , their embeddings,  $\mathbf{x}$ , can be computed using numerous approaches such as using bi-directional LSTMs (Melamud et al., 2016) or transformers (Reimers and Gurevych, 2019). In our experiments, we use the Smoothed Inversed Frequency (SIF; Arora et al., 2017), which computes document embeddings as the weighted-average of the pre-trained word embeddings for the words in a document. Despite being unsupervised, SIF has shown strong performance in numerous semantic textual similarity benchmarks (Agirre et al., 2015). Using the centroid computed in (1), similarity for target instance to the centroid is computed using

the cosine similarity given in (2).

$$\text{sim}(\mathbf{x}, \mathbf{c}_T) = \frac{\mathbf{x}^\top \mathbf{c}_T}{\|\mathbf{x}\| \|\mathbf{c}_T\|} \quad (2)$$

Other distance measures such as the Euclidean distance can also be used. We use cosine similarity here for its simplicity. We predict the labels for the target domain unlabelled instances,  $\mathcal{T}^U$ , using  $f^*$ , and select the instances with the top- $k$  highest similarities to the target domain according to (2) as the target domain’s pseudo-labelled instances  $\mathcal{T}^{L*}$ .

### 3.3 Relatedness Map Learning

Not all of the source domain instances are relevant to a given target domain instance and the performance of a classifier under domain shift can be upper bounded by the  $\mathcal{H}$ -divergence between a source and a target domain (Kifer et al., 2004; Ben-David et al., 2006, 2009). To model the relatedness between a target domain instance and each instance from the  $N$  source domains, we use the pseudo-labelled target domain instances  $\mathcal{T}^{L*}$  and source domains’ labelled instances  $\mathcal{S}_i^L$  to learn a *relatedness map*,  $\psi_i$ , between a target domain instance  $\mathbf{x}_T$  ( $\in \mathcal{T}^{L*}$ ) and a source domain labelled instance  $\mathbf{x}_i^L$  ( $\in \mathcal{S}_i^L$ ) as given by (3).

$$\psi_i(\mathbf{x}_T, \mathbf{x}_i^L) = \frac{\exp(\mathbf{x}_T^\top \mathbf{x}_i^L)}{\sum_{x' \in \mathcal{S}_i^L} \exp(\mathbf{x}_T^\top \mathbf{x}') \quad (3)}$$

Using  $\psi_i$ , we can determine how well each instance in a source domain contributes to the prediction of the label of a target domain’s instance.

### 3.4 Instance-based Domain-Attention

To avoid negative transfer, we dynamically select the source domain(s) to use when predicting the label for a given target domain instance. Specifically, we learn *domain-attention*,  $\theta(\mathbf{x}_T, \mathcal{S}_i)$ , for each source domain,  $S_i$ , conditioned on  $x_T$  as given by (4).

$$\theta(\mathbf{x}_T, \mathcal{S}_i) = \frac{\exp(\mathbf{x}_T^\top \phi_i)}{\sum_{j=1}^N \exp(\mathbf{x}_T^\top \phi_j)} \quad (4)$$

$\phi_i$  can be considered as a *domain embedding* for  $S_i$  and has the same dimensionality as the instance embeddings. During training we initialise  $\phi_i$  using Xavier initialisation (Glorot and Bengio, 2010) and normalise such that  $\forall \mathbf{x}_T, \sum_{i=1}^N \theta(\mathbf{x}_T, \mathcal{S}_i) = 1$ .

### 3.5 Training

We combine the relatedness map (§ 3.3) and domain-attention (§ 3.4) and predict the label,  $\hat{y}(x_T)$ , of a target domain instance  $x_T$  using (5).

$$\hat{y}(x_T) = \sigma \left( \sum_{i=1}^N \sum_{\mathbf{x}_i^L \in \mathcal{S}_i^L} y(x_i^L) \psi_i(\mathbf{x}_T, \mathbf{x}_i^L) \theta(\mathbf{x}_T, \mathcal{S}_i) \right) \quad (5)$$

Here,  $\sigma(z) = 1/(1 + \exp(-z))$  is the logistic sigmoid function and  $y(x_i^L)$  is the label of the source domain labelled instance  $x_i^L$ . First, we use the target instances,  $x \in \mathcal{T}^{L*}$ , with inferred labels  $y^*(x)$  (computed using  $f^*$  from Algorithm 1) as the training instances and predict their labels,  $\hat{y}(x)$ , by (5). The cross entropy error,  $E(\hat{y}(x), y^*(x))$  for this prediction is given by (6):

$$E(\hat{y}(x), y^*(x)) = -\lambda(x)(1-y^*(x))\log(1-\hat{y}(x)) - \lambda(x)y^*(x)\log(\hat{y}(x)) \quad (6)$$

Here,  $\lambda(x)$  is a rescaling factor computed using the normalised similarity score as in (7):

$$\lambda(x) = \frac{\text{sim}(\mathbf{x}, \mathbf{c}_T)}{\sum_{\mathbf{x}' \in \mathcal{T}^{L*}} \text{sim}(\mathbf{x}', \mathbf{c}_T)} \quad (7)$$

We minimise (6) using ADAM (Kingma and Ba, 2015) for learning the domain-embeddings,  $\phi_i$ . The initial learning rate is set to  $10^{-3}$  using a subset of  $\mathcal{T}^{L*}$  held-out as a validation dataset.

## 4 Experiments

To evaluate the proposed method, we use the multi-domain Amazon product review dataset compiled by Blitzer et al. (2007). This dataset contains product reviews from four domains: Books (**B**), DVD (**D**), Electronics (**E**) and Kitchen Appliances (**K**). Following Guo et al. (2018), we conduct experiments under two different splits of this dataset as originally proposed by Blitzer et al. (2007) (Blitzer2007) and by Chen et al. (2012) (Chen2012). Table 1 shows the number of instances in each dataset. By using these two versions of the Amazon review dataset, we can directly compare the proposed method against relevant prior work. Next, we describe how the proposed method was trained on each dataset.

For Blitzer2007, we use the official train and test splits where each domain contains 1600 labelled training instances (800 positive and 800 negative), and 400 target test instances (200 positive and 200

negative). In addition, each domain also contains 6K-35K unlabelled instances. We use 300 dimensional pre-trained GloVe embeddings (Pennington et al., 2014) following prior work (Bollegala et al., 2011; Wu and Huang, 2016) with SIF to create document embeddings for the reviews.

In Chen2012, each domain contains 2000 labelled training instances (1000 positive and 1000 negative), and 2000 target test instances (1000 positive and 1000 negative). The remainder of the instances are used as unlabelled instances (ca. 4K-6K for each domain). We use the publicly available<sup>3</sup> 5000 dimensional tf-idf vectors produced by Zhao et al. (2018). We use a multilayer perceptron (MLP) with an input layer of 5000 dimensions and 3 hidden layers with 500 dimensions. We use final output layer with 500 dimensions as the representation of an instance.

For each setting, we follow the standard input representation methods as used in prior work. It also shows the flexibility of the proposed method to use different (embedding vs. BoW) text representation methods. We conduct experiments for cross-domain sentiment classification with multiple sources by selecting one domain as the target and the remaining three as sources. The statistics for the two settings are shown in Table 1.

### 4.1 Effect of Self-Training

As described in § 3.1, our proposed method uses self-training to generate pseudo-labels for the target domain unlabelled instances. In Table 2, we compare self-training against alternative pseudo-labelling methods on Chen2012: Self-Training (**Self**; Abney, 2007; Chattopadhyay et al., 2012), Union Self-Training (**uni-Self**; Aue and Gamon, 2005), Tri-Training (**Tri**; Zhou and Li, 2005) and Tri-Training with Disagreement (**Tri-D**; Søgaard, 2010). We observe that all semi-supervised learning methods improve only slightly over **uni-MS**, the baseline model trained on the union of all sources and tested directly on a target domain without any DA, which has been identified as a strong baseline for multi-source DA (Aue and Gamon, 2005; Zhao et al., 2018; Guo et al., 2018). Therefore, pseudo-labelling step alone is insufficient for DA. Moreover, we observe that all semi-supervised methods perform comparably.

<sup>3</sup><https://github.com/KeiraZhao/MDAN/>

Target	Source	Train	Test	Unlabel	Train	Test	Unlabel
		Blitzer2007 (Blitzer et al., 2006)			Chen2012 (Chen et al., 2012)		
B	D,E,K	1600 × 3	400	6000	2000 × 3	2000	4465
D	B,E,K	1600 × 3	400	34741	2000 × 3	2000	5586
E	B,D,K	1600 × 3	400	13153	2000 × 3	2000	5681
K	B,D,E	1600 × 3	400	16785	2000 × 3	2000	5945

Table 1: Number of train, test and unlabelled instances for the two Amazon product review datasets.

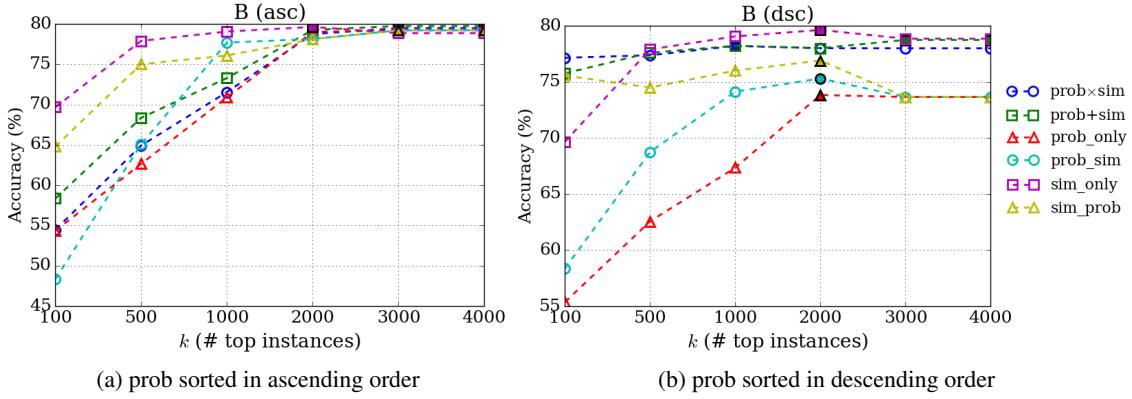


Figure 1: The number of selected pseudo-labelled instances  $k$  on **Blitzer2007** is shown on the x-axis. prob denotes prediction confidence from the pseudo classifier trained on the source domains, sim denotes the similarity to the target domain, asc and dsc respectively denote sorted in ascending and descending order (only applied to prob related selection methods, sim is always sorted in dsc). prob\_only denotes using only prediction confidence, sim\_only denotes using only target similarity. prob\_sim indicates selecting by prob first and then sim (likewise for sim\_prob). prob×sim denotes using the product of prob and sim, and prob+sim denotes using their sum. The marker for the best result of each method is filled.

Example (1) *Why anybody everest feet would want reading this? ... pure pleasure why 29028 feet account this?... It's a pleasure to read.*

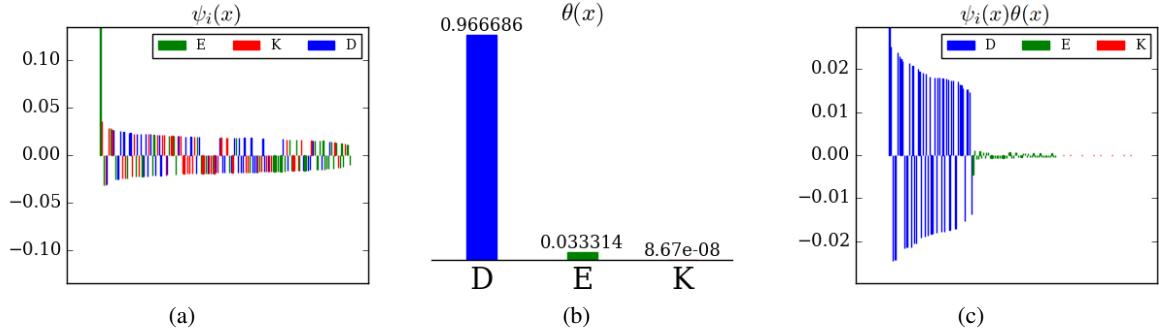


Figure 2: A positively labelled a target test instance in **B** (top) and resulted  $\theta$ ,  $\psi_i$  and the product of  $\psi_i$  and  $\theta$  (bottom). Here, the x-axis represents the instances and the y-axis represents the prediction scores. Instance specific values in (a) and (c) are shown as  $> 0$  for positive labelled instances and otherwise  $< 0$ . Source instances from **D**, **E** and **K** are shown in blue, green and red respectively. The contributions from top-150 instances from three source domains are shown.

## 4.2 Pseudo-labelled Instances Selection

When selecting the pseudo-labelled instances from the target domain for training a classifier for the target domain, we have two complementary strategies: (a) select the most confident instances according to  $f^*$  (denoted by *prob*) or (b) select the most sim-

ilar instances to the target domain’s centroid (denoted by *sim*). To evaluate the effect of these two strategies and their combinations (i.e *prob+sim* and *prob×sim*), in Figure 1, we select target instances with each strategy and measure the accuracy on the target domain **B** for increasing numbers of in-

Example (2) *Her relationship limited own pass her own analysis, there're issues mainly focus in turn for codependency. Disappointing, dysfunctional. Mother'll book her daughter's turn the pass, message turn the message issues analysis of very disappointing information.*

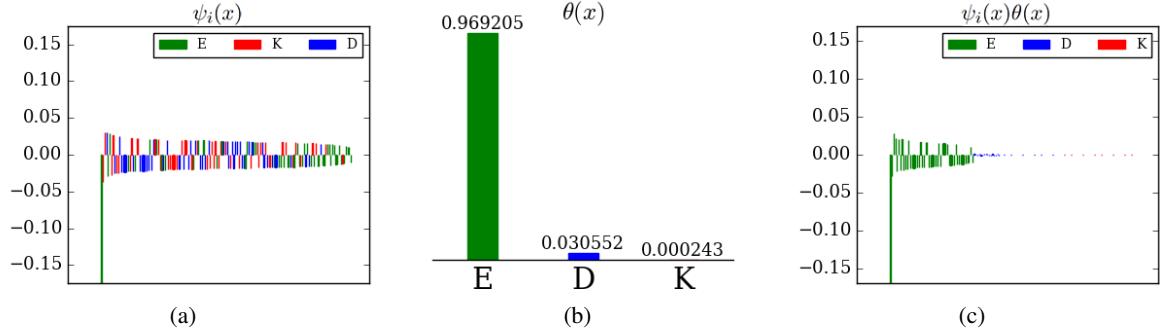


Figure 3: A negatively labelled target test instance in **B**.

DM	L	Score	Evidences (Reviews)
E	-	0.16943	Serious problems.
E	-	0.02823	Sound great but lacking isolation in other areas.
E	+	0.02801	Cases for the cats walking years, no around and knocking...walking on similar cases of cats.
E	+	0.02233	Cord supposed to no problems, this extension extension not worked as cord did...whatever expected just worked fine.
E	-	0.02209	Buy this like characters not used names...be aware of many commonly used characters before you accept file like drive.

Table 3: The top-5 evidences for Example (2) selected from the source domains. DM denotes the domain of the instance. L denotes the label for the instance. Score is  $\psi_i(x)\theta(x)$ .

T	uni-MS	Self	uni-Self	Tri	Tri-D
B	79.46	79.60	79.46	<b>79.61</b>	79.51
D	82.32	<b>82.49</b>	82.35	82.35	82.35
E	84.93	84.97	84.93	<b>84.99</b>	84.93
K	87.17	87.18	87.17	87.15	<b>87.23</b>

Table 2: Classification accuracies (%) for semi-supervised methods on **Chen2012**.

stances  $k$  in the descending (dsc) and ascending (asc) order of the selection scores.

From Figure 1b we observe that selecting the highest confident instances does not produce the best UDA accuracies. In fact, merely selecting instances based on confidence scores only (corresponds to prob\_only) reports the worst performance. On the other hand, instances that are highly similar to the target domain’s centroid are more effective for DA. We observe that with only  $k = 1000$  instances, sim\_only reaches almost its optimal accuracy. Using validation data, we estimated that  $k = 2000$  to be sufficient for all domains to reach the peak performance regardless of the selection strategy. Therefore, we selected 2000 pseudo-labelled instances for the attention step. In our

experiments, we used sim\_only to select pseudo-labelled instances because it steadily improves the classification accuracy with  $k$  for all target domains, and is competitive against other methods.

### 4.3 Effect of the Relatedness Map

In Table 4 we report the classification accuracy on the test instances in the target domain over the different steps: **uni-MS** (no adapt baseline), **Self** (self-training), **PL** (pseudo-labelling) and **Att** (attention). We use the self-training method described in Algorithm 1. The results clearly demonstrate a consistent improvement over all the steps in the proposed method. For **Self** step, the proposed method improves the accuracy only slightly without any information from the target domain. In the **PL** step, we report the results of a predictor trained on target pseudo-labelled instances. We report the evaluation results for the trained attention model in **Att**.

In **Att** step, we use the relatedness map  $\psi_i$  to express the similarity between a target instance and each of source domain instances, and the domain attention score  $\theta$  to express the relation between a target instance and each of the source domain instances. Two example test instances (one positive and one negative) from the target domain **B**

T	uni-MS	Self	PL	Att
B	79.46	79.60	79.57	<b>79.68</b>
D	82.32	82.49	82.71	<b>82.96</b>
E	84.93	84.97	<b>85.30</b>	<b>85.30</b>
K	87.17	87.18	87.30	<b>87.48</b>

Table 4: Classification accuracies (%) across different steps of the proposed method, evaluated on **Chen2012**.

are shown in Figures 2 and 3. We observe that different source instances contribute to the predicted labels in different ways. As expected, in Figure 2a more positive source instances are selected using the relatedness map for a positive target instance, and Figure 3a more negative source instances are selected for a negative target instance. After training, we find that the proposed method identifies the level of importance of different source domains. Example (1) is closer to **D**, whereas Example (2) is closer to **E** with a very high value of  $\theta$ . Figures 2c and 3c show that the instance specific contribution to the target instance. The proposed method also identifies the level of importance within the most relevant source domain. Figure 3 shows the actual reviews as the top-5 evidences from the source domains in Example (2). Negative labelled source training instance from **E**: “*Serious problem.*” is the most important instance with the highest contribution of  $\psi_i(x)\theta(x)$  to the decision.

#### 4.4 Comparisons against Prior Work

Table 5 compares the proposed method against the following methods on **Blitzer2007** dataset.

**SCL:** Structural Correspondence Learning (Blitzer et al., 2006, 2007) is a single-source DA method, trained on the union of all source domains and tested on the target domain. We report the published results from Wu and Huang (2016).

**SFA:** Spectral Feature Alignment (Pan et al., 2010) is a single-source DA method, trained on the union of all source domains, and tested on the target domain. We report the published results from Wu and Huang (2016).

**SST:** Sensitive Sentiment Thesaurus (Bollegala et al., 2011; Bollegala et al., 2013) is the SoTA multi-source DA method on **Blitzer2007**. We report the published results from Bollegala et al. (2011).

**SDAMS:** Sentiment Domain Adaptation with Multiple Sources proposed by Wu and Huang (2016). We report the results from the original paper.

T	uni-MS	SCL	SFA	SST	SDAMS	AMN	Proposed
B	80.00	74.57	75.98	76.32	78.29	79.75	<b>83.50</b>
D	76.00	76.30	78.48	78.77	79.13	79.83	<b>80.50</b>
E	74.75	78.93	78.08	83.63*	<b>84.18**</b>	80.92*	80.00*
K	85.25	82.07	82.10	85.18	<b>86.29</b>	85.00	86.00

Table 5: Classification accuracies (%) for the proposed method and prior work on **Blitzer2007**. Statistically significant improvements over **uni-MS** according to the Binomial exact test are shown by “\*” and “\*\*” respectively at  $p = 0.01$  and  $p = 0.001$  levels.

T	uni-MS	mSDA	DANN	MDAN	MoE	Proposed
B	79.46	76.98	76.50	78.63	79.42	<b>79.68</b>
D	82.32	78.61	77.32	80.65	<b>83.35</b>	82.96
E	84.93	81.98	83.81	85.34	<b>86.62</b>	85.30
K	86.71	84.26	84.33	86.26	<b>87.96</b>	87.48

Table 6: Classification accuracies (%) for the proposed method and prior work on **Chen2012**.

**AMN:** End-to-End Adversarial Memory Network (Li et al., 2017) is a single-source DA method, trained on the union of all source domains, and tested on the target domain. We report the published results from Ding et al. (2019).

In Table 6, we compare our proposed method against the following methods on **Chen2012**.

**mSDA:** Marginalized Stacked Denoising Autoencoders proposed by Chen et al. (2012). We report the published results from Guo et al. (2018).

**DANN:** Domain-Adversarial Neural Networks proposed by Ganin et al. (2016). We report the published results from Zhao et al. (2018).

**MDAN:** Multiple Source Domain Adaptation with Adversarial Learning proposed by Zhao et al. (2018). We report the published results from the original paper.

**MoE:** Mixture of Experts proposed by Guo et al. (2018). We report the published results from the original paper.

From Tables 5 and 6, we observe that the proposed method obtains the best classification accuracy on Books domain (**B**) in both settings, which is the domain with the smallest number of unlabelled instances. In particular, when the amount of training instances are small, pseudo-labelling and domain-attention in our proposed method play a vital role in multi-source UDA. Although **SDAMS** (in **Blitzer2007**) and **MoE** (in **Chen2012**) outperform the proposed method, the simplicity and the ability to provide explanations are attractive properties for a UDA method when applying in an industrial setting involving a massive number of

source domains such as sentiment classification in E-commerce reviews.

## 5 Conclusions

We propose a multi-source UDA method that combines self-training with an attention module. In contrast to prior works that select pseudo-labelled instances based on prediction confidence of a predictor learnt from source domains, our proposed method uses similarity to the target domain during adaptation. Our proposed method reports competitive performance against previously proposed multi-source UDA methods on two splits on a standard benchmark dataset.

## References

- Steven Abney. 2007. *Semisupervised Learning for Computational Linguistics*, 1st edition. Chapman & Hall/CRC.
- Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Iñigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, German Rigau, Larraitz Uria, and Janyce Wiebe. 2015. SemEval-2015 task 2: Semantic textual similarity, English, Spanish and pilot on interpretability. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 252–263, Denver, Colorado. Association for Computational Linguistics.
- Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A simple but tough-to-beat baseline for sentence embeddings. In *Proceedings of 5th International Conference on Learning Representations (ICLR)*.
- Anthony Aue and Michael Gamon. 2005. Customizing sentiment classifiers to new domains: A case study. In *Proceedings of recent advances in natural language processing (RANLP)*, volume 1, pages 2–1. Citeseer.
- Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2009. A theory of learning from different domains. *Machine Learning*, 79:151–175.
- Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. 2006. Analysis of representations for domain adaptation. In *NIPS 2006*.
- John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, Bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (ACL)*, pages 440–447, Prague, Czech Republic. Association for Computational Linguistics.
- John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 120–128, Sydney, Australia. Association for Computational Linguistics.
- D. Bollegala, D. Weir, and J. Carroll. 2013. Cross-domain sentiment classification using a sentiment sensitive thesaurus. *IEEE Transactions on Knowledge and Data Engineering*, 25(8):1719–1731.
- Danushka Bollegala, David Weir, and John Carroll. 2011. Using multiple sources to construct a sentiment sensitive thesaurus for cross-domain sentiment classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL)*, pages 132–141, Portland, Oregon, USA. Association for Computational Linguistics.
- Karsten M Borgwardt, Arthur Gretton, Malte J Rasch, Hans-Peter Kriegel, Bernhard Schölkopf, and Alex J Smola. 2006. Integrating structured biological data by kernel maximum mean discrepancy. *Bioinformatics*, 22(14):e49–e57.
- Rita Chattopadhyay, Qian Sun, Wei Fan, Ian Davidson, Sethuraman Panchanathan, and Jieping Ye. 2012. Multisource domain adaptation and its application to early detection of fatigue. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 6(4):18.
- Minmin Chen, Zhixiang Xu, Kilian Q. Weinberger, and Fei Sha. 2012. Marginalized denoising autoencoders for domain adaptation. In *Proceedings of the 29th International Conference on International Conference on Machine Learning (ICML)*, pages 1627–1634, USA. Omnipress.
- Xia Cui, Frans Coenen, and Danushka Bollegala. 2017. Tsp: Learning task-specific pivots for unsupervised domain adaptation. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*, pages 754–771.
- Xiao Ding, Qiankun Shi, Bibo Cai, Ting Liu, Yanyan Zhao, and Qiang Ye. 2019. Learning multi-domain adversarial neural networks for text classification. *IEEE Access*.
- Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030.
- Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the 13th international conference on artificial intelligence and statistics (ICAI)*, pages 249–256.

- Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th International Conference on International Conference on Machine Learning (ICML)*, pages 513–520, USA. Omnipress.
- Han Guo, Ramakanth Pasunuru, and Mohit Bansal. 2020. Multi-source domain adaptation for text classification via distancenet-bandits. In *Proc. AAAI Conference on Artificial Intelligence*.
- Jiang Guo, Darsh Shah, and Regina Barzilay. 2018. Multi-source domain adaptation with mixture of experts. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4694–4703. Association for Computational Linguistics.
- Daniel Kifer, Shai Ben-David, and Johannes Gehrke. 2004. Detecting change in data streams. In *Proceedings of the Thirtieth International Conference on Very Large Data Bases - Volume 30*, VLDB ’04, pages 180–191. VLDB Endowment.
- Young-Bum Kim, Karl Stratos, and Dongchan Kim. 2017. Domain attention with an ensemble of experts. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 643–653, Vancouver, Canada. Association for Computational Linguistics.
- Diederik P. Kingma and Jimmy Lei Ba. 2015. Adam: A method for stochastic optimization. In *Proc. of ICLR*.
- Zheng Li, Yu Zhang, Ying Wei, Yuxiang Wu, and Qiang Yang. 2017. End-to-end adversarial memory network for cross-domain sentiment classification. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2237–2243. AAAI Press.
- Oren Melamud, Jacob Goldberger, and Ido Dagan. 2016. context2vec: Learning generic context embedding with bidirectional LSTM. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 51–61, Berlin, Germany. Association for Computational Linguistics.
- Tanmoy Mukherjee, Makoto Yamada, and Timothy Hospedales. 2018. Learning unsupervised word translations without adversaries. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 627–632.
- Sinno Jialin Pan, Xiaochuan Ni, Jian-Tao Sun, Qiang Yang, and Zheng Chen. 2010. Cross-domain sentiment classification via spectral feature alignment. In *Proceedings of the 19th International Conference on World Wide Web (WWW)*, WWW ’10, pages 751–760, New York, NY, USA. ACM.
- Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359.
- Jeffery Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: global vectors for word representation. In *Proceedings of Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.
- Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3980–3990, Hong Kong, China. Association for Computational Linguistics.
- Ryan Rifkin and Aldebaro Klautau. 2004. In defense of one-vs-all classification. *Journal of Machine Learning Research*, 5:101–141.
- Michael T Rosenstein, Zvika Marx, Leslie Pack Kaelbling, and Thomas G Dietterich. 2005. To transfer or not to transfer. In *Proceedings of Conference on Neural Information Processing Systems (NeurIPS) workshop on Inductive Transfer: 10 Years Later*, volume 898, pages 1–4.
- Sebastian Ruder and Barbara Plank. 2018. Strong baselines for neural semi-supervised learning under domain shift. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1044–1054. Association for Computational Linguistics.
- Lex Razoux Schultz, Marco Loog, and Peyman Mohajerin Esfahani. 2018. Distance based source domain selection for sentiment classification. *arXiv preprint arXiv:1808.09271*.
- Anders Søgaard. 2010. Simple semi-supervised training of part-of-speech taggers. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 205–208. Association for Computational Linguistics.
- Sainbayar Sukhbaatar, arthur szlam arthur arthur arthur, Jason Weston, and Rob Fergus. 2015. End-to-end memory networks. In *Advances in Neural Information Processing Systems 28*, pages 2440–2448. Curran Associates, Inc.
- Fangzhao Wu and Yongfeng Huang. 2016. Sentiment domain adaptation with multiple sources. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, volume 1, pages 301–310.
- Kun Zhang, Mingming Gong, and Bernhard Scholkopf. 2015. Multi-source domain adaptation: A causal view. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI)*, pages 3150–3157. AAAI Press.
- Han Zhao, Shanghang Zhang, Guanhong Wu, José MF Moura, Joao P Costeira, and Geoffrey J Gordon. 2018. Adversarial multiple source domain adaptation. In *Advances in Neural Information Processing Systems*, pages 8568–8579.

Zhi-Hua Zhou and Ming Li. 2005. Tri-training: exploiting unlabeled data using three classifiers. *IEEE Transactions on Knowledge and Data Engineering*, 17(11):1529–1541.

# Compressing Pre-trained Language Models by Matrix Decomposition

Matan Ben Noach<sup>†</sup> and Yoav Goldberg<sup>\*‡</sup>

<sup>\*</sup>Computer Science Department, Bar-Ilan University, Ramat-Gan Israel

<sup>†</sup>Intel AI Lab, Petah-Tikva Israel

<sup>‡</sup>Allen Institute for Artificial Intelligence

matan.ben.noach@intel.com, yoav.goldberg@gmail.com

## Abstract

Large pre-trained language models reach state-of-the-art results on many different NLP tasks when fine-tuned individually; They also come with a significant memory and computational requirements, calling for methods to reduce model sizes (green AI). We propose a two-stage model-compression method to reduce a model’s inference time cost. We first decompose the matrices in the model into smaller matrices and then perform feature distillation on the internal representation to recover from the decomposition. This approach has the benefit of reducing the number of parameters while preserving much of the information within the model. We experimented on BERT-base model with the GLUE benchmark dataset and show that we can reduce the number of parameters by a factor of 0.4x, and increase inference speed by a factor of 1.45x, while maintaining a minimal loss in metric performance.

## 1 Introduction

Deep learning models have been demonstrated to achieve state-of-the-art results, but require large parameter storage and computation. It’s estimated that training a Transformer model with a neural architecture search has a  $CO_2$  emissions equivalent to nearly five times the lifetime emissions of the average U.S. car, including its manufacturing (Strubell et al., 2019). Alongside the increase in deep learning models complexity, in the NLP domain, there has been a shift in the NLP modeling paradigm from training a randomly initialized model to fine-tuning a large and computational heavy pre-trained language model (Howard and Ruder, 2018; Peters et al., 2018; Devlin et al., 2018; Radford, 2018; Radford et al., 2019; Dai et al., 2019; Yang et al., 2019; Lample and Conneau, 2019; Liu et al., 2019b; Raffel et al., 2019; Lan et al., 2019; Lewis et al., 2019).

While re-using pre-trained models offsets the training costs, inference time costs of the fine-tuned models remain significant, and are showstoppers in many applications. The main challenge with pre-trained models is how can we reduce their size while saving the information contained within them. Recent work, approached this by keeping some of the layers while removing others (Sanh et al., 2019; Sun et al., 2019; Xu et al., 2020). A main drawback of such approach is in its coarse-grained nature: removing entire layers might discard important information contained within the model, and working at the granularity of layers makes the trade-off between compression and accuracy of a model hard to control. Motivated by this, in this work we suggest a more fine-grained approach which decomposes each matrix to two smaller matrices and then perform feature distillation on the internal representation to recover from the decomposition. This approach has the benefit of preserving much of the information while reducing the number of parameters. Alongside the advantage of preserving the information within each layer, there is also a memory flexibility advantage compared to removing entire layers; As a result of decomposing each matrix to two smaller matrices, we can store each of the two matrices in two different memory blocks. This has the benefit of distributing the model matrices in many small memory blocks, which is useful when working in shared CPU-based environments.

We evaluated our approach on the General Language Understanding Evaluation (GLUE) benchmark dataset (Wang et al., 2018) and show that our approach is superior or competitive in the different GLUE tasks to previous approaches which remove entire layers. Furthermore, we study the effects of different base models to decompose and show the superiority of decomposing a fine-tuned model compared to a pre-trained model or a ran-

domly initialized model. Finally, we demonstrate the trade-off between compression and accuracy of a model.

## 2 Related Work

In the past year, there have been many attempts to compress transformer models involving pruning (McCarley, 2019; Guo et al., 2019; Wang et al., 2019; Michel et al., 2019; Voita et al., 2019; Gordon et al., 2020), quantization (Zafir et al., 2019; Shen et al., 2019) and distillation (Sanh et al., 2019; Zhao et al., 2019; Tang et al., 2019; Mukherjee and Awadallah, 2019; Sun et al., 2019; Liu et al., 2019a; Jiao et al., 2019; Izsak et al., 2019). Specifically, works on compressing pre-trained transformer language models focused on pruning layers. Sun et al. (2019) suggested to prune layers while distilling information from the unpruned model layers. Xu et al. (2020) proposed to gradually remove layers during training.

We also note that very recently a work similar to ours was uploaded to arxiv (Mao et al., 2020). There are a few differences from their work to ours. Firstly, we distill different parts of the model (see Section 3 for details). Secondly, we focus on training the decomposed model and do not prune the model parameters. Thirdly, our base model, which is used for decomposition and as a teacher, is a fine-tuned model; This has the benefit of task-specific information as we show in our experiments in Section 4.2.

## 3 Method

Our goal is to decompose each matrix  $W \in R^{n \times d}$  as two smaller matrices, obtaining an approximated matrix  $W' = AB$ ,  $A \in R^{n \times r}$ ,  $B \in R^{r \times d}$ , where  $r < \frac{nd}{n+d}$ . We seek a decomposition s.t.  $W'$  is close to  $W$  in the sense that  $d(Wx, W'x)$  is small for all  $x$ , where  $d$  is a distance metric between vectors. In practice, we require the condition to hold not for all  $x$ , but for vectors seen in a finite relevant sample (in our case, the training data). While one could start with random matrices and optimize the objective using gradient descent, we show that a two-staged approach performs better: we first decompose the matrices using SVD, obtaining  $A'$ ,  $B'$  s.t.  $\|A'B' - W\|_2^2$  is small (SVD is guaranteed to produce the best rank- $r$  approximation to  $W$ , (Stewart, 1991)). We then use these matrices as initialization and optimize  $d(Wx, W'x)$  (feature distillation), while

also optimizing for task loss. We show that this process works substantially better in practice. Our loss function is thus composed of three different objectives:

**Cross Entropy Loss** The cross entropy loss over an example  $x$  with label  $y$  is defined likewise:  $L_{CE} = -\log p_s(y|x)$ , where  $p_s$  is the probability for label  $y$  given by the decomposed student model.

**Knowledge Distillation Loss** The goal of knowledge distillation is to imitate the output layer of a teacher model by a student model. The Knowledge Distillation Loss is defined likewise:  $L_{KD} = \left\| \frac{\mathbf{z}_s - \mathbf{z}_t}{T} \right\|_2$ , where  $\mathbf{z}_s$  and  $\mathbf{z}_t$  are the logits of the decomposed and original models respectively and  $T$  is a temperature hyper-parameter.

**Feature Distillation Loss** The goal of feature distillation is to imitate the intermediate layers of a teacher model by a student model. we use the following intermediate representations to distill the knowledge from<sup>1</sup>:

- Query, Key and Value Layers - The dot product of a matrix of concatenated tokens representation vectors  $X$  by the query, key and value parameter matrices,  $Z_q = X \cdot W^Q$ ,  $Z_k = X \cdot W^K$ ,  $Z_v = X \cdot W^V$
- Attention Matrix - The attention matrix probabilities.  $Z_{att} = softmax(Z_q \cdot Z_k^T)$
- Attention Heads - The output of the attention heads.  $Z_H = Z_{att} \cdot Z_v$
- The Multihead Attention Layer Output - The dot product of the attention heads by the matrix  $W^O$ .  $Z_{MH} = Z_H \cdot W^O$
- The first feed forward layer - The dot product of the multihead attention layer by the first feed forward layer.  $Z_{f1} = Z_{MH} \cdot W_1$
- The second feed forward layer - The dot product of the first feed forward layer by the second feed forward layer.  $Z_{f2} = Z_{f1} \cdot W_2$

We denote  $S_z^i$  and  $T_z^i$  as the intermediate representations which were described above of layer  $i$  for

<sup>1</sup>We follow the notations of Vaswani et al. (2017) for the transformer parameters and omit biases for notation convenience.

the decomposed student and original teacher models respectively. Our loss function then is defined

$$\text{by: } L_{FD} = \sum_i^{\mathcal{T}_z^i, S_z^i} \sum_{T_z, S_z} \|T_z - S_z\|_2$$

**Full Objective** Our loss function is then defined by a weighted combination of these three loss functions likewise:  $L = \alpha L_{CE} + (1 - \alpha)L_{KD} + L_{FD}$  where  $\alpha \in [0, 1]$  is a chosen hyper-parameter.

## 4 Experiments

We compare various variants of our compression method, corresponding to different subsets of our loss. All variants decompose the matrices using SVD, but differ in their objective functions. These correspond to the four last lines in Table 1. Low Rank BERT Fine-tuning (LRBF) corresponds to  $L = L_{CE}$ . LRBF+KD corresponds to  $L = \alpha L_{CE} + (1 - \alpha)L_{KD}$ . LRBF+FD corresponds to  $L = L_{CE} + L_{FD}$ , while LRBF+FD+KD corresponds to the complete objective.

The other lines in the table correspond to uncompressed model (first line) and to baselines which prune layers and distill. Fine-tuning fine-tunes a six layered BERT model. Vanilla KD trains a six-layered BERT model with  $L = \alpha L_{CE} + (1 - \alpha)L_{KD}$ . BERT-PKD trains a six layered BERT model with  $L = \alpha L_{CE} + (1 - \alpha)L_{KD}$  while also adding an  $L_{FD}$  objective, but on the hidden states between every consecutive layer. BERT-of-Theseus fine-tunes BERT model while gradually pruning half of the layers. We chose this baselines for several reasons: like our method they result in a practical reduction of parameters;<sup>2</sup> they are task-specific;<sup>3</sup> and they do not require the pre-training stage, which is expensive and not practical for most practitioners.

**Datasets** We evaluate our proposed approach on the General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2018), a collection of diverse NLP tasks.

**Training Details** We fine-tune a pre-trained BERT model (Devlin et al., 2018) for each task with a batch size of 8 and a learning rate of  $2e-5$  for 3 epochs with an early stop mechanism according to the validation set. We perform the matrix

<sup>2</sup>Unlike, e.g., pruning, which sets parameters to zero and requires specialized hardware to fully take advantage of.

<sup>3</sup>Unlike, e.g., DistillBERT which is meant to be run before fine-tuning.

decomposition on every parametric weight matrix of the encoder (excluding the embedding matrix) in a fine-tuned model and train the decomposed model as the student model and the original fine-tuned model as the teacher. For each task we train for 3 epochs with an early stopping mechanism according to the task validation set, the maximum sequence length is 128 and we perform a grid search over the learning rates  $\{2e^{-6}, 5e^{-6}, 2e^{-5}, 5e^{-5}, 2e^{-4}, 5e^{-4}\}$  and 5 different seeds and choose the best model according to the validation set of each task.<sup>4</sup> For knowledge distillation hyper-parameters we used a temperature hyper-parameter  $T = 10$  and  $\alpha = 0.7$ .<sup>5</sup>

### 4.1 Main Results

Table 1 compares the results for validation and test of other compression approaches which prune layers, along with low rank models which were fine-tuned and trained with one or more of the distillation objectives described in Section 3. As can be seen, Low Rank BERT Feature Distillation + KD and Low Rank BERT Feature Distillation surpass all of results of all methods in both validation and test sets except BERT-of-Theseus method in the test set, in which Low Rank BERT Feature Distillation + KD surpasses the results in 5 of the tasks and reach comparable results in 2 of the tasks. Also, as can be seen knowledge distillation alone is not sufficient to compensate for the decomposition, but it slightly improves the results when incorporating feature distillation alone.

### 4.2 Further Analysis

**Effect of Base Model and Decomposition** In this experiment we test the importance of the base model we use to decompose and use as a teacher. We compared between three types of distillation sources: fine-tuned teacher, pre-trained teacher and no teacher. Furthermore, we compared between three types of model initializations: a decomposed fine-tuned model, a decomposed pre-trained model and a randomly initialized model with the same architecture as the decomposed models. The results are shown in Table 2, on all tasks when training with no teacher distilla-

<sup>4</sup>We detailed the changes we made to the original fine-tuning procedure, every other hyper-parameters which were not mentioned, is set as described in (Devlin et al., 2018).

<sup>5</sup>We chose those hyper-parameters from a grid search over  $T = \{5, 10, 20\}$  and  $\alpha = \{0.2, 0.5, 0.7\}$  on the MRPC validation set.

Method	CoLA		MNLI		MRPC		QNLI		QQP		RTE		SST-2		STS-B		Macro Score	
	Dev	Test																
<i>Uncompressed Models - 110M Parameters</i>																		
<i>6 Layers Transformer Models - 66M Parameters</i>																		
BERT-base (uncompressed)	59.9	53.9	84.6	83.9	89.0	85.6	91.6	90.9	88.1	80.2	71.5	67.2	93.5	93.6	89.8	84.8	83.5	80.0
Fine-tuning	43.4	41.5	80.1	80.1	86.0	83.1	86.9	86.7	87.8	78.7	62.1	63.6	89.6	90.7	81.9	81.1	77.2	75.7
Vanilla KD (Hinton et al., 2015)	45.1	42.9	80.1	80.0	86.2	83.4	88.0	88.3	88.1	79.5	64.9	64.7	90.5	91.5	84.9	81.2	78.5	76.4
BERT-PKD (Sun et al., 2019)	45.5	43.5	81.3	81.3	85.7	82.5	88.4	89.0	88.4	79.8	66.5	65.5	91.3	92.0	86.2	82.5	79.2	77.0
BERT-of-Theseus (Xu et al., 2020)	51.1	<b>47.8</b>	82.3	82.3	89.0	85.4	89.5	89.6	89.6	<b>80.5</b>	68.2	66.2	91.5	92.2	88.7	<b>84.9</b>	81.2	78.6
<i>Low Rank Approximated Models - 65.2M Parameters (This Work)</i>																		
Low Rank BERT Fine-tuning	41.0	40.5	82.9	82.3	82.4	79.8	89.4	88.8	89.0	79.5	65.0	60.4	91.3	92.0	87.0	81.2	78.5	75.6
Low Rank BERT + KD	44.7	34.0	83.1	82.4	83.4	80.4	89.1	88.7	89.0	79.9	64.3	60.6	91.3	91.5	86.6	80.9	78.9	74.8
Low Rank BERT Feature Distillation	<b>51.2</b>	43.4	<b>84.9</b>	<b>83.8</b>	<b>89.4</b>	<b>86.1</b>	<b>91.4</b>	<b>90.7</b>	<b>89.8</b>	<b>80.5</b>	<b>70.8</b>	<b>66.0</b>	<b>92.2</b>	<b>92.9</b>	<b>89.3</b>	<b>84.2</b>	<b>82.4</b>	<b>78.4</b>
Low Rank BERT Feature Distillation + KD	53.0	42.9	84.8	83.7	90.4	86.2	91.4	90.8	89.7	80.5	71.1	67.8	92.4	92.9	89.4	84.6	<b>82.8</b>	<b>78.7</b>

Table 1: Results on GLUE dev and test sets. Metrics are *Accuracy* (MNLI (average of MNLI match and MNLI mis-match), QNLI, RTE, SST-2), *Avg of Accuracy and F1* (MRPC, QQP), *Matthew’s correlation* (CoLA), *Avg of Pearson and Spearman correlations* (STS-B). BERT-base (Teacher) is our fine-tuned BERT model. The numbers for the 6 layered models are taken from (Xu et al., 2020), Best results are indicated in Bold.

Base Model/Teacher Model	CoLA			MRPC			SST-2		
	Fine-tuned	Pre-trained	None	Fine-tuned	Pre-trained	None	Fine-tuned	Pre-trained	None
Fine-tuned	$48.7 \pm 2.4$	$47.5 \pm 0.7$	$40.1 \pm 0.6$	$88.5 \pm 0.5$	$85.8 \pm 0.5$	$81.6 \pm 1.3$	$91.8 \pm 0.4$	$91.3 \pm 0.5$	$90.9 \pm 0.4$
Pre-trained	$49.4 \pm 1.7$	$44.8 \pm 2.1$	$10.8 \pm 2.6$	$89.2 \pm 0.4$	$86.3 \pm 1.0$	$77.1 \pm 0.6$	$91.7 \pm 0.2$	$91.2 \pm 0.4$	$89.6 \pm 1.1$
Random	$3.6 \pm 5.1$	$0.0 \pm 0.0$	$0.6 \pm 0.6$	$75.9 \pm 1.1$	$75.3 \pm 0.7$	$75.0 \pm 0.4$	$88.2 \pm 0.5$	$87.2 \pm 0.7$	$81.2 \pm 0.5$

Table 2: Results on the dev set of CoLA, MRPC and SST-2 tasks with different initializations and different teachers. The results are averages and standard deviations of five runs with different seeds.

tion, the results are best when decomposing a fine-tuned model and decomposing a pre-trained model is better than randomly initializing a model; This indicates that the decomposition saves the information within the model and when decomposing a fine-tuned model it saves some of the more task specific information. Furthermore, on all tasks and all initialization the best results are when using a fine-tuned model as a teacher.

Rank (Parameter Count)	CoLA	MRPC	SST-2
Full Rank (110M)	$58.4 \pm 1.2$	$88.3 \pm 0.7$	$92.8 \pm 0.5$
350 (82.6M)	$57.7 \pm 0.9$	$88.9 \pm 0.7$	$92.0 \pm 0.5$
245 (65.2M)	$48.7 \pm 2.4$	$88.5 \pm 0.5$	$91.8 \pm 0.4$
150 (49.4M)	$38.7 \pm 1.6$	$87.8 \pm 0.6$	$91.3 \pm 0.4$

Table 3: Results on the dev set of CoLA, MRPC and SST-2 tasks with different ranks. The results are averages and standard deviations of five runs with different seeds.

**Compression vs. Performance Trade-off** Our method requires to determine a rank for the compression. But can we achieve better results when choosing a higher rank? Can we choose a lower rank for smaller models and still achieve satisfactory results? To determine this we experimented on three different ranks. As shown in Table 3, higher ranks achieve better results, while lower ranks achieve satisfactory results while compromising metric performance.

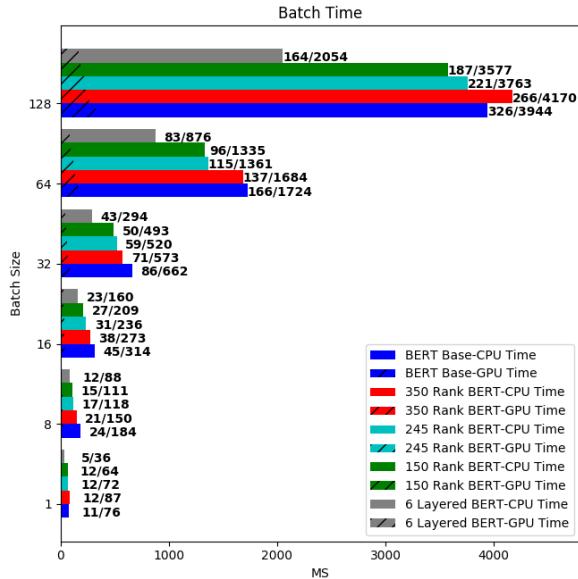


Figure 1: Average time in milliseconds to run a batch of samples from all of the GLUE tasks, when running on a Intel(R) Xeon(R) Platinum 8180 CPU @ 2.50GHz and on a single TITAN V 12GB GPU.

**Run-time Savings** In this experiment we measured the average time in milliseconds it takes for BERT-base compared to its decomposed and six-layered counterparts to output predictions for a batch of samples with varying batch sizes. As shown in Figure 1, we still gain a significant time performance improvement when running on both CPU and GPU architectures over a BERT-base

model. Models that are decomposed to a rank  $r = 245$  are about 1.45 faster than their uncompressed counterpart for batches larger than one when running on a GPU and around  $1.2 - 1.55$  faster for batches 8, 16, 32, 64 when running on a CPU. Furthermore, higher ranks still benefit running time and lower ranks improve the running time further. Also, we note that although a six-layered BERT does achieve faster inference time, due to the coarse-grained compression, it losses more information contained within it and thus achieves inferior results; As shown in the results in Table 1, a six-layered model trained with distillation (e.g. BERT-PKD (Sun et al., 2019)) achieves significantly lower results and the BERT-of-Theseus model, which does improve upon BERT-PKD, requires many training iterations to achieve this to overcome the loss of information when gradually removing entire layers, which result in higher training times.

## 5 Conclusions

We presented a way to compress pre-trained large language models fine-tuned for specific tasks, while preserving much of the information contained within them, by using matrix decomposition to two small matrices. For future work it might be interesting to combine this approach with another approach such as pruning or quantization to achieve smaller models.

## Acknowledgements

This project has received funding from the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme, grant agreement No. 802774 (iEXTRACT) and was sponsored in part by an Intel AI grant to the Bar-Ilan University NLP lab.

## References

- Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G. Carbonell, Quoc Viet Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2978–2988.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805.
- Mitchell A. Gordon, Kevin Duh, and Nicholas Andrews. 2020. Compressing BERT: studying the effects of weight pruning on transfer learning. *CoRR*, abs/2002.08307.
- Fu-Ming Guo, Sijia Liu, Finlay S. Mungall, Xue Lin, and Yanzhi Wang. 2019. Reweighted proximal pruning for large-scale language representation. *CoRR*, abs/1909.12486.
- Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. *ArXiv*, abs/1503.02531.
- Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 328–339.
- Peter Izsak, Shira Guskin, and Moshe Wasserblat. 2019. Training compact models for low resource entity tagging using pre-trained language models. *CoRR*, abs/1910.06294.
- Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. Tinybert: Distilling BERT for natural language understanding. *CoRR*, abs/1909.10351.
- Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *CoRR*, abs/1901.07291.
- Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A lite BERT for self-supervised learning of language representations. *CoRR*, abs/1909.11942.
- Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2019. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461.
- Linqing Liu, Huan Wang, Jimmy Lin, Richard Socher, and Caiming Xiong. 2019a. Attentive student meets multi-task teacher: Improved knowledge distillation for pretrained models. *CoRR*, abs/1911.03588.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.
- Yihuan Mao, Yujing Wang, Chufan Wu, Chen Zhang, Yang Wang, Yaming Yang, Quanlu Zhang, Yunhai Tong, and Jing Bai. 2020. Ladabert: Lightweight adaptation of BERT through hybrid model compression. *CoRR*, abs/2004.04124.
- J. S. McCarley. 2019. Pruning a bert-based question answering model. *CoRR*, abs/1910.06360.

- Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? *CoRR*, abs/1905.10650.
- Subhabrata Mukherjee and Ahmed Hassan Awadallah. 2019. Distilling transformers into simple neural networks with unlabeled transfer data. *CoRR*, abs/1910.01769.
- Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke S. Zettlemoyer. 2018. Deep contextualized word representations. *ArXiv*, abs/1802.05365.
- Alec Radford. 2018. Improving language understanding by generative pre-training.
- Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683.
- Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.
- Sheng Shen, Zhen Dong, Jiayu Ye, Linjian Ma, Zhewei Yao, Amir Gholami, Michael W. Mahoney, and Kurt Keutzer. 2019. Q-BERT: hessian based ultra low precision quantization of BERT. *CoRR*, abs/1909.05840.
- G. W. Stewart. 1991. Perturbation theory for the singular value decomposition. *SVD and Signal Processing, II: Algorithms, Analysis and Applications*, pages 99–109.
- Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 3645–3650.
- Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for BERT model compression.
- Raphael Tang, Yao Lu, Linqing Liu, Lili Mou, Olga Vechtomova, and Jimmy Lin. 2019. Distilling task-specific knowledge from BERT into simple neural networks. *CoRR*, abs/1903.12136.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5998–6008.
- Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5797–5808.
- Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the Workshop: Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@EMNLP 2018, Brussels, Belgium, November 1, 2018*, pages 353–355.
- Ziheng Wang, Jeremy Wohlwend, and Tao Lei. 2019. Structured pruning of large language models. *CoRR*, abs/1910.04732.
- Canwen Xu, Wangchunshu Zhou, Tao Ge, Furu Wei, and Ming Zhou. 2020. Bert-of-theseus: Compressing BERT by progressive module replacing. *CoRR*, abs/2002.02925.
- Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237.
- Ofir Zafrir, Guy Boudoukh, Peter Izsak, and Moshe Wasserblat. 2019. Q8BERT: quantized 8bit BERT. *CoRR*, abs/1910.06188.
- Sanqiang Zhao, Raghav Gupta, Yang Song, and Denny Zhou. 2019. Extreme language model compression with optimal subwords and shared projections. *CoRR*, abs/1909.11687.

# You May Like This Hotel Because ...: Identifying Evidence for Explainable Recommendations

Shin Kanouchi<sup>1</sup>, Masato Neishi<sup>2</sup>, Yuta Hayashibe<sup>1</sup>, Hiroki Ouchi<sup>3</sup>, Naoaki Okazaki<sup>4</sup>

<sup>1</sup>Megagon Labs, Tokyo, <sup>2</sup>The University of Tokyo,

<sup>3</sup>RIKEN, <sup>4</sup>Tokyo Institute of Technology

{shin187nlp, hayashibe}@megagon.ai,

neishi@tkl.iis.u-tokyo.ac.jp,

hiroki.uchi@riken.jp, okazaki@c.titech.ac.jp

## Abstract

Explainable recommendation is a good way to improve user satisfaction. However, explainable recommendation in dialogue is challenging since it has to handle natural language as both input and output. To tackle the challenge, this paper proposes a novel and practical task to explain evidences in recommending hotels given vague requests expressed freely in natural language. We decompose the process into two subtasks on hotel reviews: *evidence identification* and *evidence explanation*. The former predicts whether or not a sentence contains evidence that expresses why a given request is satisfied. The latter generates a recommendation sentence given a request and an evidence sentence. In order to address these subtasks, we build an Evidence-based Explanation dataset, which is the largest dataset for explaining evidences in recommending hotels for vague requests. The experimental results demonstrate that the BERT model can find evidence sentences with respect to various vague requests and that the LSTM-based model can generate recommendation sentences.

## 1 Introduction

Recently, dialog systems using Natural Language Processing technology have been adopted in interactive services such as call centers (Zumstein and Hundertmark, 2017). One challenging issue in a real-world scenario is vague requests<sup>1</sup> from users. For example, in a hotel booking service, users often ask operators for “a child-friendly hotel” or “a convenient inn.” To respond to such vague requests, human operators need to explain the reason why the given request

is satisfied. An example response would be, “This hotel has a large kids’ space, so I recommend it for families with children like you.” Responding to vague requests with evidences is effective because it not only strengthens the recommendation, but also urges users to make more concrete requests such as “I don’t need a kids’ space but want a baby stroller rental service.”

Several studies have addressed explainable recommendations that produce natural language sentences (Zhao et al., 2014; Zhang et al., 2014; Wang et al., 2018; Zhao et al., 2019). One major approach is feature-based explanations. Zhang et al. (2014) generated explanation sentences using templates with slots, for example, “You might be interested in [feature], on which this product performs well.” However, by handling only predefined and limited features, this study cannot explain detailed evidences for each hotel such as “a view of Mount Fuji and Lake Kawaguchi.” Furthermore, this study does not accept natural language requests as inputs, which is a major bottleneck for building dialog-based interactive systems.

In this study, we propose a novel and practical task to identify and explain evidences that satisfy a given vague request expressed freely in natural language. Specifically, assuming a practical situation of recommendation, we address a hotel booking service. When choosing a hotel on an interactive service, users make a wide range of vague requests, which differ from predefined aspects (Wang et al., 2010), emotional expressions (Chen et al., 2010) and questions (Rajani et al., 2019). In order to satisfy vague requests by recommending hotels with evidences, the system must understand a given request, associate the request to a hotel with spe-

<sup>1</sup>In this study, a vague request means one that does not specify a specific product, experience or service.

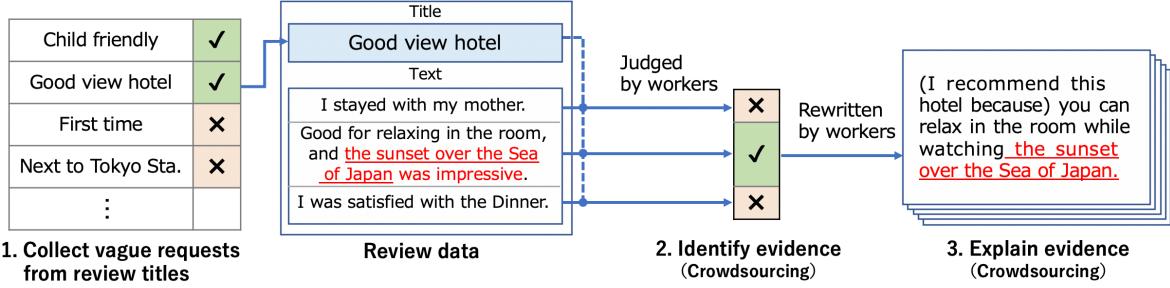


Figure 1: Pipeline for building the Evidence-based Explanation dataset

cific evidence, and generate an explanation (recommendation sentence) for the evidence.

To address these challenges, we decompose the process into two subtasks: *Evidence Identification* and *Evidence Explanation*. The former predicts whether a sentence contains evidence that expresses why a given request is satisfied. The latter generates a recommendation sentence given the evidence sentence. In order to focus on evidence explanations for requests, we assume that recommending hotels are given in advance in this study.

For these subtasks, we present an Evidence-based Explanation dataset, which is the largest dataset for explaining evidences in recommending hotels for vague requests. Assuming that titles of hotel reviews often correspond to vague requests, the dataset includes 37,280 hotel reviews with annotations for vague requests, evidence sentences for the requests, recommendation sentences based on the evidence sentences. The key feature of the dataset is the variety of requests: it includes 15,767 unique types of requests written in natural language. This dataset is publicly available<sup>2</sup>.

We report experiments for the two subtasks in Section 3. We build a BERT (Devlin et al., 2019) model for the first subtask, which predicts whether a sentence contains evidence for a request. Experimental results show that the model can detect evidence sentence for various requests with a high (79.94) F1-score, and that the score does not drop so much even for requests unseen in the training data. We present encoder-decoder models for the second subtask, which rewrite an evidence sentence into a recommendation sentence. The experiments demonstrate that an LSTM (Luong et al., 2015) based model achieves the BLEU score (Papineni et al., 2002) of 56.09 with a gold evidence sentence given and that of 45.38 without a gold sentence (only a re-

view and a request is given). We also report experiments when the two subtasks are combined to generate a recommendation sentence for a given review.

The contributions of this paper are as follows:

1. We propose a novel and practical task to explain evidences given vague requests expressed freely in natural language.
2. We create a new dataset by annotating review sentences with evidences and rewriting each evidence into a recommendation sentence. This is the largest dataset for explaining evidences in recommending hotels for vague requests.
3. Experiments show that our dataset enables to train models that can effectively find evidences to various vague requests and generate recommendation sentences.

## 2 Dataset Creation

In this section, we describe the procedure to create the Evidence-based Explanation dataset. The dataset is expected to include (i) vague requests from users, (ii) items (in this study, hotel candidates), (iii) evidence where an item satisfies an request, (iv) and a recommendation sentence based on each evidence. As a corpus that meets these requirements, we use review data on Jalan<sup>3</sup>, which is a major hotel booking service in Japan.

On jalan, users can enter reviews after their stay at the hotel. In addition to review texts, Jalan accepts ratings for some specific aspects (e.g., ‘Service’ and ‘Cleanliness’), similarly to other booking services (Wang et al., 2010). Although some aspects are similar to vague requests(e.g., “good service” or “cheap hotel”), the number of such pre-

<sup>2</sup><https://github.com/megagonlabs/ebe-dataset> <sup>3</sup><https://www.jalan.net/>

Category	Examples of requests		# of collection		# of annotations
	With “inn” or “hotel”	Additional titles	“inn”	Additional	(Types)
Clean	Clean hotel	Clean	15k	71k	3.6k (0.8k)
Relax	Relaxing inn	Grate place to relax	8k	80k	3.6k (1.1k)
Service	Helpful hotel	staff were very helpful	10k	143k	3.4k (2.0k)
Useful	Useful inn	Useful for sightseeing	4k	113k	2.7k (1.1k)
Child friendly	Child friendly hotel	Child friendly	3k	81k	2.6k (1.3k)
Good view	Good view hotel	Good view	1k	34k	2.3k (0.7k)
Delicious	Hotel with delicious food	Delicious dinner	1k	145k	2.3k (1.0k)
Cost	Good low cost hotel	Low cost but very good	5k	89k	2.2k (1.3k)
Good	Perfect hotel	Perfect	33k	278k	2.6k (1.4k)
Others	Historic hotel	Historic atmosphere	19k	297k	11.8k (5.2k)
Total	—	—	99k	1.3M	37.3k (15.8k)

Table 1: Examples of collected vague requests and the number of collections, uses, and types

defined aspects is very limited and cannot cover diverse requests, such as “dog-friendly hotel.”

Consequently, we created a new dataset using review titles and review texts. In the review texts, users describe their impressions on the service of the hotel based on their real experiences. Additionally, the review titles often summarize the most salient point of the experiences and often include similar expressions to vague requests such as “dog-friendly hotel.” Hence, assuming that some review titles express vague requests and that the corresponding review texts contain evidence, we extracted vague requests from review titles and annotated evidence sentences for requests in review texts. Finally, we rewrote the evidence sentences into recommendation sentences.

Figure 1 illustrates the overall pipeline to construct the dataset. It consists of three steps.

- 1. Collect vague requests from review titles:** Use rules to find review titles that correspond to vague requests.
- 2. Identify evidence:** Ask crowdworkers to identify whether each review sentence contains evidence for the request corresponding to the review title.
- 3. Explain evidence:** Ask crowdworkers to write recommendation sentences based on the evidence sentences.

## 2.1 Collecting Vague Requests

Based on the fact that some titles have similar expressions to vague requests, we collected vague requests by selecting review titles. Some review titles are inappropriate as requests, for example, “Thanks” or “Stayed for the first time.” Therefore, to comprehensively collect vague requests for hotels with less noise, we first extracted review titles

that included words representing accommodations such as “inn” or “hotel.” In addition, we applied filtering rules to remove other unuseful titles<sup>4</sup>.

Considering the possibility of data imbalance, we performed a categorical analysis. First, we applied morphological analysis of the collected requests using SudachiPy (Takaoka et al., 2018) to normalize surface variations in the requests. We manually checked and categorized all filtered titles appearing more than twenty times in the corpus, which resulted in ten categories of vague requests.

The distribution of categories in the dataset was skewed; the numbers of instances for some categories were small. For example, “Good hotel” is common but not “Hotel with delicious food.” This is because a small percentage of requests appear with the expression “inn” or “hotel.” Titles such as “Delicious dinner” are more frequent than “Hotel with delicious food.” Therefore, we extracted additional titles that contained the same content words as the extracted titles, excluding the accommodation expressions such as “inn” or “hotel.” For example, “Hotel with delicious food” → “delicious” (excluding hotel and extracting a content word) → “Delicious dinner” (additional titles).

Table 1 shows examples of vague requests collected from review titles. We extracted about 1.4 million reviews (99k + 1.3M) that have the collected requests in titles (# of collection). For annotation in the next subsection, we selected 37,280 reviews (# of annotations). By expanding the collection rules, the number of requests increased greatly, and the data imbalance problem reduced. Furthermore, it also increased the variation of the request expressions. Overall, we collected 15,767 unique kinds of titles in 37,280 reviews.

<sup>4</sup>The rules include, for example, titles must not contain proper nouns and must contain one or more content words.

	Clean	Relax	Service	Useful	Child	View	Delicious	Cost	Good	Others	All
Ratio of Relevant [%]	82.1	69.7	85.3	83.8	71.9	82.6	91.6	69.6	72.9	68.6	75.6
Ratio of Evidence [%]	48.3	55.4	71.4	74.1	58.8	60.1	68.6	44.3	56.0	46.1	55.3

Table 2: Ratio of relevant and evidence sentences included in the review text for a request

Amount of evidence sentences	# of reviews
No evidence	16,654 (44.7%)
1 evidence sentence	16,456 (44.1%)
2 evidence sentences	3,382 (9.1%)
$\geq 3$ evidence sentences	788 (2.1%)

Table 3: Amount of evidence sentences in each review

## 2.2 Evidence Identification Dataset

We used Yahoo Crowdsourcing<sup>5</sup> to annotate review data with evidence for requests. Workers were shown a review title and a single sentence of the review text. Then they were asked, “Is the following sentence relevant to the title, and does it contain evidence for the title?” There were three options for the answer: Evidence, Relevant (not as Evidence), and Irrelevant. Relevant (not as Evidence) means that the sentence contains the same expression as the request or its synonymous expression, but it does not present an evidence to support the request (title). Although the evidence may make sense by combining two or more sentences, we annotated each sentence of the review independently to simplify the annotation work. We annotated 37,280 reviews in total (“# of annotations” in Table 1). For a higher quality, each task was annotated by five people. We also prepared check questions for each task.

Table 2 reports the ratios of the Evidence and Relevant instances by category. In the ‘Useful’ category, 74% of the reviews contained evidence in the text, while only 44% of the reviews in the ‘Cost’ category did. This is because users apt to explain the reason for an ‘useful’ hotel in a review, but because the necessity of explaining the reason for ‘cheap’ hotel is relatively low.

Table 3 shows the number of evidence sentences for each review request. Approximately half of the reviews contained evidence. Requests that have a lot of evidence per review were an unique feature of this dataset. For example, requests that express

<sup>5</sup>It is a microtask crowdsourcing service in Japan. We mixed some check questions in the tasks and receive annotated data from only workers who answered the check questions correctly. We did not set gender or attribute limits of workers in all our tasks. <https://crowdsourcing.yahoo.co.jp/>

general goodness such as “good hotel” have lots of evidence. In this case, the task of labeling evidence sentences was similar to annotation efforts for sentiment analysis.

## 2.3 Evidence Explanation Dataset

Using crowdsourcing, we rewrote evidence sentences into recommendation sentences. First, we showed workers a review title and an evidence sentence. Then we asked them to write a recommendation sentence so that the sentence can be used to explain the evidence in recommending the hotel to a user. We annotated 25,804 sentences that at least three of the five workers judged to contain evidence in Section 2.2. We asked workers to report the following two cases. (1) The request is a negative expression such as “bad view.” (2) There is no evidence in a given sentence<sup>6</sup>. To ensure the quality of the annotation, each sentence was annotated by five workers, and we prepared check questions for each task. In the check questions, we prepared negative expressions for requests, and confirmed that the workers followed the instructions properly.

Table 4 shows the number of the exact matches of five workers for the created recommendation sentence. When only extracting a phrase from a review is sufficient as a recommendation sentence, the five workers tended to produce an identical result. On the other hand, when a certain part in a review had to be rewritten, recommendation sentences from the five workers tended to differ.

## 3 Experiments

Using the annotated dataset, we conducted two experiments. (1) Evidence Identification and (2) Evidence Explanation. The former predicts whether a sentence contains evidence for a request, whereas the latter generates a recommendation sentence.

<sup>6</sup>We targeted sentences where at least three people judged to contain evidence. However, it was sometimes difficult to write recommendation sentences when two out of five workers judged that the sentence has no evidence.

# of same answers	# of sent.	Title	Examples Evidence sentence	Recommendation sentence
$\geq 2$ matches	13,100	Pet-friendly	This hotel is tolerant of dog lovers because you can sleep in a bed with your dog.	(We recommend this) because you can sleep in a bed with your dog.
All different	9,889	Nice open-air bath	The temperature of the bath was just right, and we spent a long time in the open-air bath watching the stars.	(We recommend this) because you can take a long open-air bath while gazing at the stars.
Negative req. $(\geq 3)$	1,651	The scenery ...	We booked a Bay Bridge view, but it was only visible from the edge of the window.	—
No Evidence	1,164	Mountain side view	It was an ocean view hotel but we stayed on the mountain side.	—

Table 4: Examples of recommendation sentences rewritten by workers and matching rate of rewriting

	Reviews	Sentences	Positive (%)
Train	29,826	148,671	20,709 (13.9)
Dev	3,726	18,549	2,606 (14.0)
Test	3,728	18,823	2,489 (13.2)
Total	37,280	186,043	25,804 (13.9)

Table 5: Evaluation data for evidence identification

### 3.1 Evidence Identification Task

**Task Description** The task is to predict whether or not a sentence contains an evidence for a request. This is a binary classification problem. A positive example is a sentence to which at least three out of the five workers labeled evidence. All other sentences are treated as negative examples.

We randomly divided the data by review into training, development, and test set (see Table 5). We used the same data split in all experiments.

**Experimental Settings** We explored logistic regression<sup>7</sup> and BERT (Devlin et al., 2019) as classification models. For the tokenization, we used juman++<sup>8</sup> (Tolmachev et al., 2018) and Byte pair encoding (BPE) (Sennrich et al., 2016) with the vocabulary size of 8k. We pre-trained word2vec (Mikolov et al., 2013) CBOW model, and the BERT model on two million review sentences in Jalan. For the logistic regression, we calculated the TF-IDF<sup>9</sup> (Jones, 1972) vector and the average vector of word2vec for requests and sentences respectively. We used the request vector, the sentence vector, and the difference between the two vectors as features. The input to the BERT model was in the following order: request sentence, [SEP], and evidence sentence. Hyperparameters of each model were tuned by the F1-score on the development set.

<sup>7</sup>Implemented in: <https://scikit-learn.org>

<sup>8</sup><https://github.com/ku-nlp/jumanpp>

<sup>9</sup>We used the word frequency in the sentence as TF, and the word frequency of the review text as DF.

**Results and Analysis** Table 6 reports F1-score of both models for each category and all categories. The F1-score of BERT for all the data was 79.94, which is 33.15 points higher than the logistic regression. Results in each category show that BERT had the highest F1-score for ‘Useful’ and the lowest for ‘Good’. We analyze the results of the evidence identification by the BERT model from different perspectives in the following paragraphs.

**Evidence Identification without Requests** The F1-score of the BERT model was relatively high, considering the nature of this task, i.e., associating evidences to requests. However, we need to make sure whether the BERT model considers a request when identifying an evidence. Thus, we trained another BERT model without a request (only a sentence is given) as an input. The model trained without a request resulted in the F1-score of 43.22, which is 37 points lower than that with a request. This huge gap indicates that evidences in our dataset depend on requests and that the BERT model pays attention to requests properly.

For example, a model trained with a request predicts that the sentence, “It was pleasant in the room with a view of the sea” is evidence for a request “good view” but not for “good food”. In contrast, a model trained without a request predicts that the both are evidence sentences.

**Unseen Requests** Since the dataset contains a wide range of requests, 30% of the requests in the test set are unseen, not appearing in the training set. Thus, we divided the test set in terms whether a request is unseen or not, and computed the F1-score in Table 7. Although the F1-score for unseen requests drops by 6.44 points, it is still high compared to the score trained without a request (described in the previous paragraph). This indicates that the model makes a successful prediction for

Model	Clean	Relax	Service	Useful	Child	View	Delicious	Cost	Good	Others	All
Logistic regression	44.39	46.03	51.21	61.30	49.39	61.02	54.34	30.24	34.76	37.15	46.79
BERT	79.52	82.89	84.98	89.48	85.04	81.23	82.54	73.59	68.85	73.89	79.94

Table 6: F1-score for evidence identification for each category

Whether a sentence contains an explicit conjunction (e.g., <b>Because</b> )				Quadrant	F1
Explicit		Implicit			
Yes (e.g., <b>Clean hotel</b> )	[+] This hotel was <b>clean because it was renovated.</b> [-] I chose a <b>clean hotel because</b> I had hay fever. Evidence ratio: 59.7% (142 / 238)	[+] The hotel was <b>renovated</b> and <b>clean.</b> [-] This hotel was cheap and <b>clean.</b> Evidence ratio: 44.1% (1,181 / 2,678)	A	Evidence ratio: 44.1% (1,181 / 2,678)	87.11
No	[+] I was satisfied <b>because it was renovated.</b> [-] I was satisfied <b>because</b> it was cheap. Evidence ratio: 8.2% (157 / 1,909)	[C] [D] [+] [+] It was recently renovated. [-] It was cheap. Evidence ratio: 7.2% (1,009 / 13,998)	B	Evidence ratio: 7.2% (1,009 / 13,998)	82.64

Figure 2: Characteristics of evidence sentences

Quadrant	F1
A	87.11
B	82.64
C	79.01
D	75.87
A+B	83.12
C+D	76.30
A+C	82.82
B+D	79.54
All	79.94

Table 8: F1-score for each quadrant

	F1	# of instances
Unseen requests	75.40	5,857
Seen requests	81.84	12,966

Table 7: F1-score for unseen/seen requests

majority of the unknown requests.

Examining successful predictions for unseen requests, we found that the same expression to the request often appears in the evidence sentence. For example, in response to a request for “*a good location to watch* a football game,” the evidence sentence includes, “It’s located in front of Tusu Station in Saga, and it’s *a good location to watch* the Tusu football game.” The expression in italic is considered to be a clue for predicting the evidence label for the sentence. The analysis of whether the request is included in the evidence sentence is discussed in detail in the next paragraph.

In contrast, we observed difficult instances as well. For example, the request (review title) is, “You can fully enjoy an *extraordinary* experience,” and the evidence sentence is, “I was refreshed by soaking in a hot spring while listening to the chirping of birds and the sound of insects.” The BERT model could not infer that the experience (hot spring, chirping birds) is *extraordinary* and that the sentence is an evidence for the request.

**Characteristics of Evidence Sentences** There are various ways to express an evidence sentence, for example, with and without a use of conjunctions. Figure 2 illustrates four categories (decomposed into two axes) of how a sentence presents an evidence for a request. The y-axis is whether a request expression appears in an evidence sentence.

The x-axis is whether there is an explicit conjunction (e.g., ‘because’) expressing the discourse relation between a request and evidence. We have automatically divided these categories by rules.

The top-left quadrant A includes a request expression and an explicit conjunction in the sentence. Although 60% contain evidence for a request, quadrant A has the smallest volume. On the other hand, the lower-right quadrant D has the largest volume, but has the smallest ratio of including evidence for the request (only 7%). The evidence for quadrant A can be collected by a simple rule, but it is comprised of only about 6% of the total evidence. Our dataset successfully extracts other evidence expressions using the relationship between the review title and the text.

Table 8 shows the F1-score of the BERT model for each quadrant. The F1-score of quadrant A, which contains an explicit conjunction and request words, was highest (87.11). It was 7.17 points higher than the average F1-score of all test data. On the other hand, the F1-score of quadrant D, which does not contain an explicit conjunction nor any request words, was lowest (75.87). It was 4.07 points lower than the average F1-score of all test data. In addition, the F1-score of quadrant A+B was 6.82 points higher than the F1-score of quadrant C+D, indicating that the presence of the request expression in the evidence sentence significantly impacts on the performance of predicting evidence.

We examined successful cases in quadrant D, which is the most difficult of all. In these cases, we found that expressions similar to the requests often appear in the evidence sentence. For exam-

ple, in response to the request ‘‘I am soothed by a *meal*,’’ the evidence sentence is ‘‘I was impressed by the deliciousness of the freshly made egg rolls for *breakfast*.’’ In the example, the word ‘meal’ in the request is related to the word ‘breakfast.’ However, the model could not recognize that the sentence, ‘‘We have a foot washing place next to the entrance, gum roller and wet tissue, it was very thorough,’’ contains an evidence for the request, ‘‘An inn where I can stay with my pet dog.’’ This may be due to the lack of similar expressions for the request in the sentence, and the failure to associate dog and dog amenities.

### 3.2 Evidence Explanation Task

**Task Description** The task generates a recommendation sentence given request and evidence sentences. We used only the data that three or more workers rewrote into recommendation sentences in Section 2.3. Each evidence sentence had multiple recommendation sentences rewritten by the workers, and we use all of them as training data. We use BLEU (Papineni et al., 2002) to evaluate generated sentences.

**Experiment Settings** We compared three models: a rule-based model and two neural network models. The rule-based model rewrites an evidence sentence into a recommendation sentence by focusing on the root node in the parse tree of the evidence sentence. The rules include: if the root node is a verb, adjective, or auxiliary verb, add ‘‘because’’ at the beginning; if the root node is a noun, add ‘‘because of’’ at the beginning; and if the root node is an adverb, add ‘‘because you can do’’ at the beginning.

For neural network models, we employed an LSTM model with attention (Luong et al., 2015) and a Transformer model (Vaswani et al., 2017), assuming that the task is translation from an evidence sentence into a recommendation sentence. We used the FAIRSEQ (Ott et al., 2019) to implement the models. We tokenized it using Juman++ and BPE. The input to the model was in the following order: request sentence, [SEP], and evidence sentence. Hyper-parameters of the models were tuned by the BLEU score on the development set.

For the evaluation, we used the BLEU score on sentences tokenized by Juman++ (not by BPE). Since the number of references for each evidence sentence was not constant, we randomly selected one.

Method	BLEU
No-rewrite	47.17
Rule-based	50.26
LSTM	56.09
Transformer	55.79

Table 9: BLEU score to generate recommendation given evidence and a request

Method	BLEU	F1
Pipeline (BERT → LSTM)	45.38	63.30
End-to-end (LSTM)	16.27	49.13

Table 10: BLEU score to generate recommendation given review text and a request

**Results and Analysis** Table 9 shows BLEU scores of generated recommendation sentences. ‘No-rewrite’ is the baseline where the evidence sentence is treated as the recommendation sentence without a rewrite. Compared with this baseline (47.17 BLEU), all generation methods obtained higher BLEU scores. The score of the LSTM-based model (56.09) was 0.30 points higher than that of the Transformer-based model (55.79). However, the BLEU score of the rule-based model was only 5.83 point lower than the LSTM-based model. This implies that this task requires fewer rewrites than we expected.

There are some differences between the outputs of the rule-based model and the LSTM-based model. The rule-based model tends to produce longer sentences because it cannot generate a sentence from scratch. In addition, the rule-based model fails when an evidence sentence includes unnecessary information, for example, ‘‘it’s close to the station and it’s convenient, so we’d like to use it again.’’ The LSTM-based model could successfully generate ‘‘(We recommend this hotel) Because it’s also close to the station and it’s convenient,’’ although the rule-base model kept ‘‘so we’d like to use it again’’ and generated, ‘‘Because it’s close to the station and it’s convenient, so we’d like to use it again.’’

### 3.3 End-to-end Experiment

In this section, we present an experiment to generate a recommendation sentence given review data (a request and review sentences) as an input. Combining the subtasks 1 and 2, this end-to-end experiment converts a hotel review into a recommendation sentence. When a given review does not have evidence, a system is expected to yield a special

<b>Review title:</b> The view was great!
<b>Review text:</b> On the first day, I purchased a premium sightseeing ticket before check-in. Then I went to the Higashiyama Zoo to meet a handsome gorilla, Shabani. On the next day, I went to the Maglev and Railway Museum, and enjoyed it. In the room, where Nagoya Castle can be seen beautifully, the children were delighted. ...
<b>Reference:</b> (I recommend this hotel) Because the room has a beautiful view of Nagoya Castle
<b>Pipeline:</b> Because the children were overjoyed because they had a beautiful view of Nagoya Castle.
<b>End-to-end:</b> Because you have a beautiful view of Nagoya Castle from your room.
<b>Review title:</b> Happy with my doggy
<b>Review text:</b> The couple, father and two dogs used it. ... The rooms were spacious, and the large windows overlooked the sea. I was very satisfied. ... While many pet-friendly hotels are disappointing, this one is recommended. ...
<b>Reference:</b> While many pet-friendly hotels are disappointing, this one is recommended
<b>Pipeline:</b> Because pet-friendly hotels are very convenient
<b>End-to-end:</b> Because the rooms were spacious the large windows overlooked the sea, and I was very satisfied
<b>Review title:</b> A hotel recommended for families with children
<b>Review text:</b> Because I have a 16-month old child, I was drawn to the plan that included room service and a private hot spring... The mattress was thin because of its age, so it would have been better if it were thicker. The hot springs and customer service were good, and it was good that the staff treated my children kindly.
<b>Reference:</b> Because the hot springs and customer service were good, and it was good for children.
<b>Pipeline:</b> Because the hot springs and customer service were good, and it was good for children.
<b>End-to-end:</b> Because the pool and customer service were good, and it was good for children.

Table 11: Examples of generating recommendation sentences given the review data

token [no-evidence].

We explored two approaches, pipeline and end-to-end. The pipeline method is simply a combination of the models from Sections 3.1 and 3.2. The method first predicts whether a sentence in a review present an evidence for a request by using the BERT model. It then generates a recommendation sentence by using the LSTM-based model for the request and the predicted evidence sentence with the highest score assigned by the BERT model only when the review includes evidence sentences. If the BERT model predicts no sentence in the review as evidence, the method generates [no-evidence].

The end-to-end method is an encoder-decoder LSTM model that directly generates a recommendation sentence given a review title and text. An input to the model is request and [SEP], followed by multiple sentences of the review. When a review did not contain an evidence for the request, the model is trained to generate [no-evidence].

Table 10 shows the BLEU scores and the macro-average F1-scores of the methods. The macro-average F1-score is defined similarly to the evaluation conducted by Rajpurkar et al. (2016)<sup>10</sup>. The pipeline method outperformed the end-to-end method, achieving a BLEU score of 45.38, 29.11 points higher than the end2end model. This is probably because the pipeline model could utilize

<sup>10</sup>The metric measures matches of bag-of-tokens in the reference and generated sentences. For reviews without an evidence, we regard that the system output is correct if the generated output is no-evidence.

the pre-trained BERT model and because training the end-to-end method was difficult with very long sequences of tokens given as inputs. In addition, the end-to-end method tends to output too many [no-evidence] and the total number of output words is low, so the BLEU score is also low due to brevity penalty.

Table 11 presents examples of the generated sentences. In the first example, the both models successfully generated appropriate recommendation sentences. Although the end-to-end method generated the natural sentence in the second example, the recommendation is nothing to do with the request, “happy with my doggy.” In the third example, the end-to-end method generated the word “pool”, which was actually false because the the review text only refers to “hot spring.” We observed these incorrect generations from the end-to-end method more than from the pipeline method.

## 4 Related Work

Several studies addressed explainable recommendations (Sarwar et al., 2001; Diao et al., 2014; Zhao et al., 2014; Zhang et al., 2014; Wang et al., 2018; Zhang et al., 2020; Zhao et al., 2019). In feature-based explanations, Zhang et al. (2014) generated textual sentences as explanations using templates such as “You might be interested in [feature], on which this product performs well.” In aspect-based explanations, Wang et al. (2010) discovered latent ratings on each aspect, and selected sentences related to each aspect to help users better understand the opinions given a set of review

texts with the overall ratings. Zhao et al. (2019) formulated a problem called personalized reason generation and generated a recommendation sentence given a song name, author, and user tag as input. The inputs of those studies were user vectors created from the user’s action history or limited aspects. However, our study deals with a wide range of natural language requests for a dialog system in the hotel booking domain.

In the field of sentiment analysis, research that extracts evidence based on sentiment expressions has attracted attention (Chen et al., 2010; Gui et al., 2016; Kim and Klinger, 2018). Chen et al. (2010) extracted the cause of a target emotional expression based on a rule. Gui et al. (2016) annotated an emotional expression and its cause. These studies aimed to gather useful information to extract emotional expressions and provide evidence simultaneously by examining the reputations for specific products. Although our study also aims to collect useful information, the requests are not limited to emotional expressions. In addition, we generate recommendation sentences.

Our study can be viewed as a special application of argument mining in the domain of hotel review. Liu et al. (2017) used manually annotated arguments of evidence-conclusion discourse relations in 110 hotel reviews. The study showed the effectiveness of several combinations of argument-based features. In Japanese, Murakami et al. (2009) proposed a method to collect consents and dissents for queries that can be answered with Yes or No. As part of that, they extracted evidence using rules. Our dataset is useful as training data to extract evidence in argument mining.

## 5 Conclusion

We proposed a novel task of predicting an evidence to satisfy a request and generating a recommendation sentence. We built an Evidence-based Explanation dataset for the task. The experimental results demonstrated that the BERT model could find evidence sentences with respect to various vague requests and that the LSTM-based model could generate recommendation sentences.

Future directions of this study include choosing the best evidence sentence from multiple candidate sentences for a vague request from a user and developing a concierge service that can recommend a hotel with evidence.

## Acknowledgments

We would like to thank the anonymous reviewers for their valuable feedback.

## References

- Ying Chen, Sophia Yat Mei Lee, Shoushan Li, and Chu-Ren Huang. 2010. Emotion cause detection with linguistic constructions. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010)*, pages 179–187.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2019)*, pages 4171–4186.
- Qiming Diao, Minghui Qiu, Chao-Yuan Wu, Alexander J Smola, Jing Jiang, and Chong Wang. 2014. Jointly modeling aspects, ratings and sentiments for movie recommendation (JMARS). In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 193–202.
- Lin Gui, Dongyin Wu, Ruifeng Xu, Qin Lu, and Yu Zhou. 2016. Event-driven emotion cause extraction with corpus construction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016)*, pages 1639–1649.
- Karen Spärck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28:11–21.
- Evgeny Kim and Roman Klinger. 2018. Who feels what and why? annotation of a literature corpus with semantic roles of emotions. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING 2018)*, pages 1345–1359.
- Haijing Liu, Yang Gao, Pin Lv, Mengxue Li, Shiqiang Geng, Minglan Li, and Hao Wang. 2017. Using argument-based features to predict and analyse review helpfulness. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)*, pages 1358–1363.
- Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP 2015)*, pages 1412–1421.
- Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Proceedings of the 1st International Conference on Learning Representations (ICLR 2013)*.

- Koji Murakami, Eric Nichols, Suguru Matsuyoshi, Asuka Sumida, Shouko Masuda, Kentaro Inui, and Yuji Matumoto. 2009. Statement map: assisting information credibility analysis by visualizing arguments. In *Proceedings of the 3rd Workshop on Information Credibility on the Web (WICOW 2009)*, pages 43–50.
- Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations (NAACL-HLT 2019)*, pages 48–53.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002)*, pages 311–318.
- Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Explain yourself! leveraging language models for commonsense reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL 2019)*, pages 4932–4942.
- Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016)*, pages 2383–2392.
- Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web (WWW 2001)*, pages 285–295.
- Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*, pages 1715–1725.
- Kazuma Takaoka, Sorami Hisamoto, Noriko Kawahara, Miho Sakamoto, Yoshitaka Uchida, and Yuji Matsumoto. 2018. Sudachi: a Japanese tokenizer for business. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, pages 2246–2249.
- Arseny Tolmachev, Daisuke Kawahara, and Sadao Kurohashi. 2018. Juman++: A morphological analysis toolkit for scriptio continua. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations (EMNLP 2018)*, pages 54–59.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS 2017)*, pages 5998–6008.
- Hongning Wang, Yue Lu, and Chengxiang Zhai. 2010. Latent aspect rating analysis on review text data: a rating regression approach. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 783–792.
- Nan Wang, Hongning Wang, Yiling Jia, and Yue Yin. 2018. Explainable recommendation via multi-task learning in opinionated text data. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 165–174.
- Yongfeng Zhang, Xu Chen, et al. 2020. Explainable recommendation: A survey and new perspectives. *Foundations and Trends in Information Retrieval*, 14(1):1–101.
- Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. 2014. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 83–92.
- Guoshuai Zhao, Hao Fu, Ruihua Song, Tetsuya Sakai, Zhongxia Chen, Xing Xie, and Xueming Qian. 2019. Personalized reason generation for explainable song recommendation. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(4):1–21.
- Xin Wayne Zhao, Yanwei Guo, Yulan He, Han Jiang, Yuexin Wu, and Xiaoming Li. 2014. We know what you want to buy: a demographic-based system for product recommendation on microblogs. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1935–1944.
- Darius Zumstein and Sophie Hundertmark. 2017. Chatbots—an interactive technology for personalized communication, transactions and services. *IADIS International Journal on WWW/Internet*, 15(1).

# A Unified Framework for Multilingual and Code-Mixed Visual Question Answering

Deepak Gupta<sup>†</sup>, Pabitra Lenka<sup>†\*</sup>, Asif Ekbal<sup>‡</sup>, Pushpak Bhattacharyya<sup>‡</sup>

<sup>†</sup>Indian Institute of Technology Patna, India

<sup>‡</sup>International Institute of Information Technology Bhubaneswar, India

<sup>†</sup>{deepak.pcs16, asif, pb}@iitp.ac.in

<sup>†</sup>pabitra.lenka18@gmail.com

## Abstract

In this paper, we propose an effective deep learning framework for multilingual and code-mixed visual question answering. The proposed model is capable of predicting answers from the questions in Hindi, English or Code-mixed (Hinglish: Hindi-English) languages. The majority of the existing techniques on Visual Question Answering (VQA) focus on English questions only. However, many applications such as medical imaging, tourism, visual assistants require a multilinguality-enabled module for their widespread usages. As there is no available dataset in English-Hindi VQA, we firstly create Hindi and Code-mixed VQA datasets by exploiting the linguistic properties of these languages. We propose a robust technique capable of handling the multilingual and code-mixed question to provide the answer against the visual information (image). To better encode the multilingual and code-mixed questions, we introduce a hierarchy of shared layers. We control the behaviour of these shared layers by an attention-based soft layer sharing mechanism, which learns how shared layers are applied in different ways for the different languages of the question. Further, our model uses bi-linear attention with a residual connection to fuse the language and image features. We perform extensive evaluation and ablation studies for English, Hindi and Code-mixed VQA. The evaluation shows that the proposed multilingual model achieves state-of-the-art performance in all these settings.

## 1 Introduction

Visual Question Answering (VQA) is a challenging problem that requires complex reasoning over visual elements to provide an accurate answer to a natural language question. An efficient VQA system can be used to build an Artificial Intelligence (AI) agent which takes a natural language question

and predicts the decision by analyzing the complex scene(s). VQA requires language understanding, fine-grained visual processing and multiple steps of reasoning to produce the correct answer. As the existing research on VQA are mainly focused on natural language questions written in English (Antol et al., 2015; Hu et al., 2017; Fukui et al., 2016; Anderson et al., 2018; Li et al., 2018; Xu and Saenko, 2016; Shih et al., 2016), their applications are often limited.

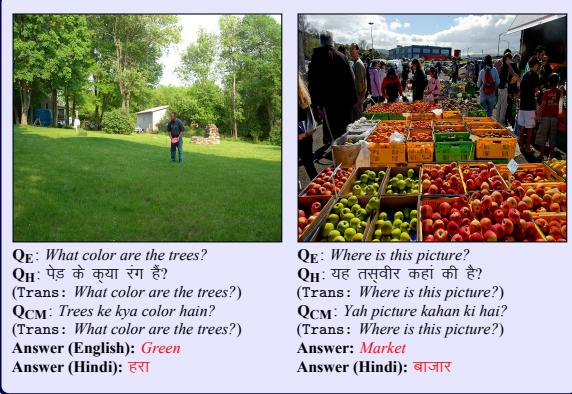


Figure 1: Examples of questions (English, Hindi and Code-mixed) with their corresponding images and answers

Multilingual speakers often switch back and forth between their native and foreign (popular) languages to express themselves. This phenomenon of embedding the morphemes, words, phrases, etc., of one language into another is popularly known as code-mixing (Myers-Scotton, 1997, 2002). Code-mixing phenomena is common in chats, conversations, and messages posted over social media, especially in bilingual / multilingual countries like India, China, Singapore, and most of the other European countries. Sectors like tourism, food, education, marketing, etc. have recently started using code-mixed languages in their advertisements to attract their consumer base. In order to build an AI agent which can serve multilingual end users,

\*Work carried out during the internship at IIT Patna

a VQA system should be put in place that would be language agnostic and tailored to deal with the code-mixed and multilingual environment. It is worth studying the VQA system in these settings which would be immensely useful to a very large number of population who speak/write in more than one language. A recent study (Parshad et al., 2016) also shows the popularity of code-mixed English-Hindi language and the dynamics of language shift in India. Our current work focuses on developing a language agnostic VQA system for Hindi, English and code-mixed English-Hindi languages.

Let us consider the examples shown in Fig 1. The majority of the VQA models (Anderson et al., 2018; Li et al., 2018; Yu et al., 2018) are capable enough to provide correct answers for English questions  $Q_E$ , but our evaluation shows that the same model could not predict correct answers for Hindi  $Q_H$  and Code-mixed question  $Q_{CM}$ . The questions  $Q_H$  and  $Q_{CM}$  correspond to the same question  $Q_E$ , but are formulated in two different languages. In this paper, we investigate the issue of multilingual and code-mixed VQA. We assume that there are several techniques available for monolingual (especially, English) VQA such that a strong VQA model can be built. However, we are interested in building a system that can answer the questions from different languages (multilingual) and the language formed by mixing up of multiple languages (code-mixed). We show that in a cross-lingual scenario due to language mismatch, applying directly a learned system from one language to another language results in poor performance. Thus, we propose a technique for multilingual and code-mixed VQA. Our proposed method mainly consists of three components. The first component is the *multilingual question encoding* which transforms a given question to its feature representation. This component handles the multilinguality and code-mixing in questions. We use multilingual embedding coupled with a hierarchy of shared layers to encode the questions. To do so, we employ an attention mechanism on the shared layers to learn language specific question representation. Furthermore, we utilize the self-attention to obtain an improved question representation by considering the other words in the question. The second component (*image features*) obtains the effective image representation from object level and pixel level features. The last component is *multimodal fusion* which is accountable to encode the question-image pair representation

by ensuring that the learned representation is tightly coupled with both the question (language) and image (vision) feature.

It is to be noted that designing a VQA system for each language separately is computationally very expensive (both time and cost), especially when multiple languages are involved. Hence, an end-to-end model that integrates multilinguality and code-mixing in its components is extremely useful. We summarize our contribution as follows:

1. We create linguistically-driven Hindi and English-Hindi code-mixed VQA datasets. To the best of our knowledge, this is the very first attempt towards this direction.
2. We propose a unified neural model for multilingual and code-mixed VQA, which can predict answer of a multilingual or code-mixed question.
3. To effectively answer a question, we enhance the vision understanding by combining local image grid and object-level visual features. We propose a simple, yet powerful mechanism based on soft-sharing of shared layers to better encode the multilingual and code-mixed questions. This bridges the gap between VQA and multilinguality.
4. We perform extensive evaluation and ablation studies for English, Hindi and Code-mixed VQA. The evaluation shows that our proposed multilingual model achieves state-of-the-art performance in all these settings.

## 2 Related Work

**Multilingual and Code-Mixing:** Recently, researchers have started investigating methods for creating tools and resources for various Natural Language Processing (NLP) applications involving multilingual (Garcia and Gamallo, 2015; Gupta et al., 2019; Agerri et al., 2014) and code-mixed languages (Gupta et al., 2018a; Bali et al., 2014; Gupta et al., 2016; Rudra et al., 2016; Gupta et al., 2014). Developing a VQA system in a code-mixed scenario is, itself, very novel in the sense that there has not been any prior research towards this direction.

**VQA Datasets:** Quite a few VQA datasets (Gao et al., 2015; Antol et al., 2015; Goyal et al., 2017; Johnson et al., 2017; Shimizu et al., 2018; Hasan et al., 2018; Wang et al., 2018) have been created to encourage multi-disciplinary research involving Natural Language Processing (NLP) and

Computer Vision. In majority of these datasets, the images are taken from the large-scale image database MSCOCO (Lin et al., 2014) or artificially constructed (Antol et al., 2015; Andreas et al., 2016; Johnson et al., 2017). There are a few datasets (Gao et al., 2015; Shimizu et al., 2018) for multilingual VQA, but these are limited only to some chosen languages, and unlike our dataset they do not offer any code-mixed challenges.

**VQA Models:** The popular frameworks for VQA in the literature are built to learn the joint representation of image and question using the attention mechanism (Kim et al., 2018; Lu et al., 2016; Yu et al., 2017; Kafle and Kanan, 2017; Zhao et al., 2017). Hu et al. (2018) proposed a technique to separately learn the answer embedding with best parameters such that the correct answer has higher likelihood among all possible answers. There are some works (Chao et al., 2018; Liu et al., 2018; Wu et al., 2018) which exploit the adversarial learning strategy in VQA. VQA has also been explored in medical domains (Zhou et al., 2018; Gupta et al., 2021; Abacha et al., 2018; Ben Abacha et al., 2019). These learned representations are passed to a multi-label classifier whose labels are the most frequent answers in the dataset. Our analysis (c.f. Section 5.5) reveals that these models perform very poorly in a cross-lingual setting.

### 3 MCVQA Dataset

**Dataset Creation:** The popular VQA dataset released by Antol et al. (2015) contains images, with their corresponding questions (in English) and answers (in English). This is a challenging large scale dataset for the VQA task. To create a comparable version of this English VQA dataset in Hindi and code-mixed Hinglish, we introduce a new VQA dataset named “Multilingual and Code-mixed Visual Question Answering” (MCVQA) which comprises of questions in Hindi and Hinglish. Our dataset<sup>1</sup>, in addition to the original English questions, also presents the questions in Hindi and Hinglish languages. This makes our MCVQA dataset suitable for multilingual and code-mixed VQA tasks. A sample of question-answer pairs and images from our dataset are shown in Fig 2.

We do not construct the answer in code-mixed language because a recent study (Gupta et al., 2018b) has shown that code-mixed sentences and

their corresponding English sentences share the same nouns (common nouns, proper nouns, spatio-temporal nouns), adjectives, etc. For example, given an English and its corresponding code-mixed question:

**Q<sub>E</sub>** : *Where is the tree in this picture?*

**Q<sub>CM</sub>** : *Is picture me tree kahan hai?*

It can be observed that both **Q<sub>E</sub>** and **Q<sub>CM</sub>** share the same noun { *picture*, *tree* }. The majority of answers in the VQA v1.0 dataset are of type ‘yes/no’, ‘numbers’, ‘nouns’, ‘verbs’ and ‘adjectives’. Therefore, we keep the same answer in both English and Code-mixed VQA dataset.

We follow the techniques similar to Gupta et al. (2018b) for our code-mixed question generation, which takes a Hindi sentence as input and generates the corresponding Hinglish sentence as the output. We translate original English questions and answers using the Google Translate<sup>2</sup> that has shown remarkable performance in translating short sentences (Wu et al., 2016). We use this service as our original questions and answers in English are very short. For the code-mixed question generation, we first obtain the Part-of-Speech<sup>3</sup> (PoS) and Named Entity<sup>4</sup> (NE) tags of each question. Thereafter, we replace the Hindi words having the PoS tags (common noun, proper noun, spatio-temporal noun, adjective) with their best lexical translation. Same strategy is also followed for the words having the NE tags as *LOCATION* and *ORGANIZATION*. The remaining Hindi words are replaced with their Roman transliteration. In order to obtain the best lexical translation, we follow the iterative disambiguation algorithm (Monz and Dorr, 2005). We generate the lexical translation by training the Statistical Machine Translation (SMT) model on the publicly available English-Hindi (EN-HI) parallel corpus (Bojar et al., 2014). Please refer to the Appendix for the comparison with other VQA datasets.

**Dataset Analysis:** The MCVQA dataset consists of 248, 349 training questions and 121, 512 validation questions for real images in Hindi and Code-mixed. For each Hindi question, we also provide its 10 corresponding answers in Hindi. In order to analyze the complexity of the generated code-mixed questions, we compute the Code-mixing Index (CMI) (Gambäck and Das, 2014) and Complexity Factor

<sup>2</sup><https://cloud.google.com/translate>

<sup>3</sup><https://bit.ly/2rpNBJR>

<sup>4</sup><https://bit.ly/2Qljan5>

<sup>1</sup>The dataset can be found here: <http://www.iitp.ac.in/~ai-nlp-ml/resources.html>

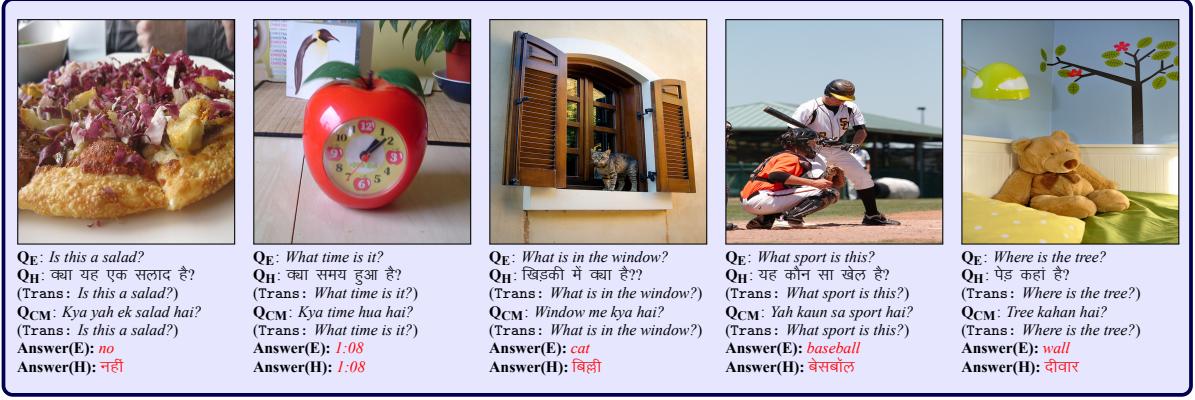


Figure 2: Sample questions (in English, Hindi and Code-Mixed) with their corresponding images and answers (in English, Hindi) from our MCVQA dataset

(CF) (Ghosh et al., 2017). These metrics indicate the level of language mixing in the questions. A detailed distribution of the generated code-mixed questions w.r.t to various metrics are in the **Appendix**.

We perform qualitative analysis by randomly selecting 5, 200 questions from our MCVQA dataset. A bilingual (En, Hi) expert was asked to manually create the code-mixed questions and translate the English questions into Hindi. We compute the BLEU (Papineni et al., 2002), ROUGE (Lin, 2004) and Translation Error Rate (TER) (Snover et al., 2006) on the human translated questions and the translations obtained from the Google Translate. We achieve high BLEU and Rouge scores (BLEU 3: 80.22; ROUGE - L: 92.20) and lower TER (9.63).

## 4 Methodology for MVQA

**Problem Statement:** Given a natural language question  $\mathcal{Q}$  in English, Hindi or code-mixed and a correlative image  $\mathcal{I}$ , the task is to perform a complex reasoning over the visual element of the image to provide an accurate natural language answer  $\hat{\mathcal{A}}$  from all the possible answers  $\mathcal{A}$ . Mathematically:

$$\hat{\mathcal{A}} = \arg \max_{\hat{\mathcal{A}} \in \mathcal{A}} p(\hat{\mathcal{A}} | \mathcal{Q}, \mathcal{I}; \phi) \quad (1)$$

where  $\phi$  is the network parameters. The architecture of our proposed methodology is depicted in Fig 3. Our proposed model has the following components:

### 4.1 Multilingual Question Encoding

Given a question<sup>5</sup>  $\mathcal{Q} = \{q_1, q_2, \dots, q_T\}$  having  $T$  words, we obtain the multilingual embedding

<sup>5</sup>It denotes the question in English, Hindi or Code-mixed

$q_t^e \in \mathbb{R}^d$  (c.f. Section 5.1) for each word  $q_t \in \mathcal{Q}$ . The resulting representation is denoted by  $\{q_t^e\}_{t=1}^T$ . We use multilingual word-embedding to obtain the lower-level representation of the words from English, Hindi and English-Hindi code-mixed questions. However, only word-embedding is not capable enough to offer multilingual and code-mixing capability. For a better multilingual and code-mixing capability at a higher level, we introduce the shared encoding layers. In order to capture the notion of a phrase, first the embedded input  $\{q_t^e\}_{t=1}^T$  is passed to a CNN layer. Mathematically, we compute inner product between the filter  $F_l \in \mathbb{R}^{l \times d}$  and the windows of  $l$  word embedding. In order to maintain the length of the question after convolution, we perform appropriate zero-padding to the start and end of the embedded input  $\{q_t^e\}_{t=1}^T$ . The convoluted feature  $q_t^{l,c}$  for  $l$  length filter is computed as follows:

$$q_t^{l,c} = \tanh(F_l q_{t+l-1}^e) \quad (2)$$

A set of filters  $L$  of different window sizes is applied on the embedded input. The final output  $q_t^c$  at a time step  $t$  is computed by the max-pooling operation over different window size filters. Mathematically,  $q_t^c = \max(q_t^{l_1,c}, q_t^{l_2,c}, \dots, q_t^{l,L,c})$ . The final representation computed by CNN layer can be denoted as  $\{q_t^c\}_{t=1}^T$ . Inspired from the success in other NLP tasks (Luong et al., 2015; Yue-Hei Ng et al., 2015), we employ stacking of multiple Bi-LSTM (Hochreiter and Schmidhuber, 1997) layers to capture the semantic representation of an entire question. The input to the first layer of LSTM is the convoluted representation of the question  $\{q_t^c\}_{t=1}^T$ .

$$q_t^r = \text{Bi-LSTM}(q_{t-1}^r, q_t^c) \quad (3)$$

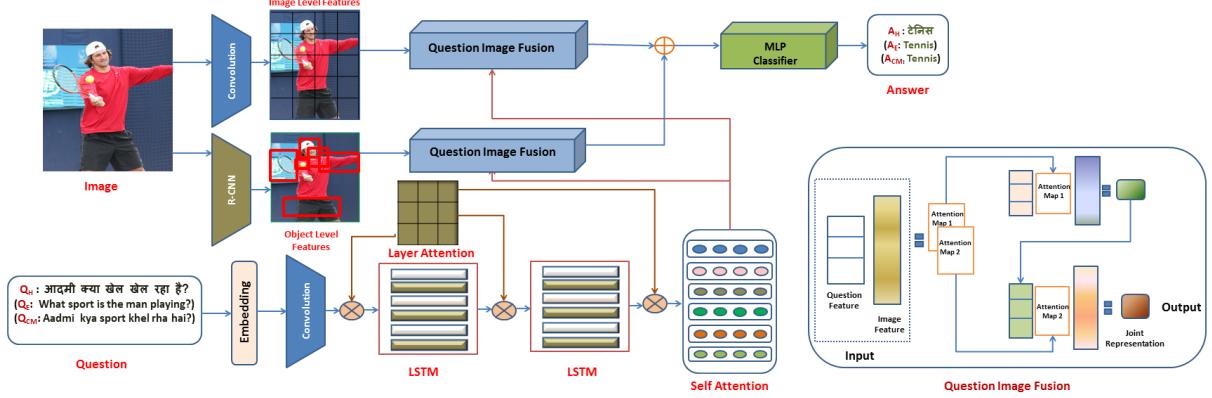


Figure 3: Architecture of the proposed multilingual VQA model. The input to the model is the multilingual question (one at a time). The bottom-right part of the image describes the *Question Image Fusion* component.

where,  $q_t^r$  and  $q_{t-1}^r$  are the hidden representations computed by the Bi-LSTM network at time  $t$  and  $t - 1$ , respectively. Specially, we compute the forward  $\overrightarrow{q}_t^r$  and backward hidden representation  $\overleftarrow{q}_t^r$  at each time step  $t$  and concatenate them to obtain the final representation  $q_t^r = \overrightarrow{q}_t^r \oplus \overleftarrow{q}_t^r$ . The output from the previous layer of LSTM is passed as input to the next layer of LSTM.

#### 4.1.1 Layer Attention

The encoding layers discussed in Section 4.1 are exploited by the questions from English, Hindi and English-Hindi code-mixed languages. It might not be the case that the representation of a question (in a given language) obtained from a particular encoding layer would also make a meaningful representation for the same question (in another language). In order to learn the language-specific control parameter for the encoding layer, we introduce an attention based mechanism over the encoding layer. Basically, our model learns an attentional vector over each encoder layer for each language. Our model learns a language importance weight matrix  $W \in \mathbb{R}^{m \times n}$ , where  $m$  and  $n$  correspond to the number of encoding layers and the number of different languages, respectively. The language importance weight matrix  $W$  is applied on a given language's ( $i$ ) question representation in the  $j^{th}$  encoding layer. Let us assume that the  $j^{th}$  multilingual encoding layer generates the question representation:  $Q^{i,j} = \{q_1^{i,j}, q_2^{i,j}, \dots, q_T^{i,j}\}$ . The language attentive representation for a language  $i$  and layer  $j$  is computed as follows:

$$\begin{aligned} \overline{q_t^{i,j}} &= \overline{W_{i,j}} q_t^{i,j}, \quad t = \{1, 2, \dots, T\} \\ \overline{W_{i,j}} &= \frac{e^{-W_{i,j}}}{\sum_{k=1}^n e^{-W_{k,j}}} \end{aligned} \quad (4)$$

The weighted question representation of  $i^{th}$  language obtained from the  $j^{th}$  layer can be denoted as  $\overline{Q^{i,j}} = \{\overline{q_1^{i,j}}, \overline{q_2^{i,j}}, \dots, \overline{q_T^{i,j}}\}$ .

In our work, we use one layer of CNN and two layers of Bi-LSTM to encode multilingual questions. At each layer of encoding, we apply language specific weight to obtain the language specific encoding layer representation. We denote the question representation obtained from the final encoding layer after applying the language specific attention as  $h = \{h_t\}_{t=1}^T$ .

#### 4.1.2 Self-Attention on Question

Inspired from the success of self-attention on various NLP tasks (Vaswani et al., 2017; Kitaev and Klein, 2018), we adopt self-attention to our model for better representation of a word by looking at the other words in the input question. The encoding obtained from multilingual encoding layer (c.f. Section 4.1.1) is passed to the self-attention layer. The multi-head self-attention mechanism (Vaswani et al., 2017) used in our model can be precisely described as follows:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_h}}\right)V \quad (5)$$

where,  $Q$ ,  $K$ ,  $V$  and  $d_h$  are the query, key, value matrices and dimension of the hidden representation obtained from the multilingual encoding layer, respectively. These matrices are obtained by multiplying different weight matrices to  $h$ . The value  $d_h$  is the dimension of the hidden representation obtained from the multilingual encoding layer. Firstly, multi-head attention linearly projects queries, keys and values to the given head ( $p$ ) using different linear projections. These projections then perform

the scaled dot-product attention in parallel. Finally, these results of attention are concatenated and once again projected to obtain a new representation. Formally, attention head ( $z_p$ ) at given head  $p$  can be expressed as follows:

$$\begin{aligned} z_p &= \text{Attention}(hW_p^Q, hW_p^K, hW_p^V) \\ &= \text{softmax}\left(\frac{(hW_p^Q)(hW_p^K)^T}{\sqrt{d_h}}\right)(hW_p^V) \end{aligned} \quad (6)$$

where  $W_p^Q$ ,  $W_p^K$  and  $W_p^V$  are the weight matrices. We exploit multiple heads to obtain the attentive representation. Finally, we concatenate all the attention heads to compute the final representation. The final question encoding obtained from the multilingual encoding layer can be represented by  $U = \{q_1^h, q_2^h, \dots, q_T^h\}$ .

## 4.2 Image Features

Unlike the previous works (Fukui et al., 2016; Yu et al., 2017; Ben-Younes et al., 2017) on VQA, in this work we extract two different levels of features, *viz.* image level and object level. We employ ResNet101 (He et al., 2016) model pre-trained on ImageNet (Deng et al., 2009) to obtain the image level features  $V_i \in \mathbb{R}^{d_i \times n_i}$ , where  $n_i$  denotes the number of spatial location of dimension  $d_i$ . We take the output of pooling layer before the final softmax layer. To generate object level features, we use the technique as discussed in Anderson et al. (2018) by using Faster R-CNN framework (Ren et al., 2017). The resulting object level features  $V_o \in \mathbb{R}^{d_o \times n_o}$  can be interpreted as ResNet features focused on the top- $n_o$  objects in the image.

## 4.3 Multimodal Fusion

We fuse the multilingual question encoding (c.f. Section 4.1.2) and image features by adopting the attention mechanism described in Kim et al. (2018). Let us denote the question encoding feature by  $U \in \mathbb{R}^{n_1 \times T}$  and the image feature by  $V \in \mathbb{R}^{n_2 \times R}$ . The  $k^{th}$  element representation using bi-linear attention network can be computed as follows:

$$f_k = (U^T X)_k^T \mathcal{M} (V^T Y)_k \quad (7)$$

where  $X \in \mathbb{R}^{n_1 \times K}$ ,  $Y \in \mathbb{R}^{n_2 \times K}$ ,  $(U^T X)_k \in \mathbb{R}^T$ ,  $(V^T Y)_k \in \mathbb{R}^R$  are the weight matrices and  $\mathcal{M} \in \mathbb{R}^{T \times R}$  is the bi-linear weight matrix. The Eq. 7 computes the 1-rank bi-linear representation of two feature vectors. We can compute the  $K$ -rank bi-linear pooling for  $f \in \mathbb{R}^K$ . With  $K$ -rank bi-linear pooling, the bi-linear feature representation

can be computed by multiplying a pooling vector  $P \in \mathbb{R}^{K \times C}$  with  $f$ .

$$\bar{f} = P^T f \quad (8)$$

where  $C$  is the dimension of the bi-linear feature vector. The  $\bar{f}$  is a function of  $U$ ,  $V$  with the parameter (attention map)  $\mathcal{M}$ . Therefore, we can represent  $\bar{f} = \text{fun}(U, V; \mathcal{M})$ . Similar to Kim et al. (2018), we compute multiple bi-linear attention maps (called as visual heads) by introducing different pooling vectors. To integrate the representations learned from multiple bi-linear attention maps, we use the multi-modal residual network (MRN) (Kim et al., 2016). Using MRN, we can compute the joint feature representation in a recursive manner:

$$\overline{f_{j+1}} = \text{fun}_j(\overline{f_j}, V; \mathcal{M}_j) \cdot \mathbf{1}^T + \overline{f_j} \quad (9)$$

The base case  $\overline{f_0} = U$  and  $\mathbf{1} \in \mathbb{R}^T$  is the vector of ones. We extract the joint feature representation for image level  $\overline{f}_i$  as well as object level feature  $\overline{f}_o$ .

## 4.4 Answer Prediction

Given the final joint representation of question with image level and object level features (c.f. Section 4.3), we augment both of these features to the counter feature ( $c_f$ ) proposed in Zhang et al. (2018). The counter feature helps the model to count the objects. Finally, we employ a two-layer perceptron to predict the answer from a fixed set of candidate answers. It is predetermined from all of the correct answers in the training set that appear more than 8 times. To this end, the logits can be computed by the following equation:

$$A_{\text{logits}} = \text{Relu}(\text{MLP}(\overline{f}_i \oplus \overline{f}_o \oplus c_f)) \quad (10)$$

The  $A_{\text{logits}}$  is passed to a  $\text{softmax}$  function to predict the answer.

## 5 Experimental Setup and Results

### 5.1 Datasets and Network Training

In our experiments, we use the VQA v1.0 dataset for English questions. There isn't a single setup for a multilingual VQA system which can handle both multilingual and code-mixed questions at the same time. Therefore, our primary motivation has been to set up a basic VQA system using the VQA v1.0 dataset. For Hindi and Code-mixed questions, we use our own multilingual VQA dataset (c.f. Section

3). Both the datasets have 248, 349 and 121, 512 questions in their training and test set, respectively. Each question has 10 answers. The test dataset of English VQA does not have publicly available ground truth answers. In order to make a fair comparison of the results in all the three setups, *viz.* English, Hindi and Code-mixed, we evaluate our proposed multilingual model on validation set of English and test set of Hindi and Code-Mixed dataset (MCVQA dataset).

The training is performed jointly with English, Hindi and Code-Mixed QA pairs by interleaving batches. We update the gradient after computing the loss of each mini-batch from a given language of sample (question, image, answer). The other baselines are trained and evaluated for each language separately. For evaluation, we adopt the accuracy metric as defined in Antol et al. (2015).

## 5.2 Hyperparameters

For English, we use the *fastText* (Bojanowski et al., 2016) word embedding of dimension 300. We use Hindi sentences from Bojar et al. (2014), and then train the word embedding of dimension 300 using the word embedding algorithm (Bojanowski et al., 2016). In order to obtain the embedding of Roman script, we transliterate<sup>6</sup> the Hindi sentence into the Roman script. These sentences are used to train the code-mixed embedding using the same embedding algorithm (Bojanowski et al., 2016), and we generate the embedding of dimension 300. These three word embeddings have the same dimensions but they are different in vector spaces. Finally, we align monolingual vectors of Hindi and Roman words into the vector space of English word embedding using the approach as discussed in Chen and Cardie (2018). While training, the model loss is computed using the categorical cross entropy function.

Optimal hyper-parameters are set to: maximum no. of words in a question=15, CNN filter size={2, 3}, # of shared CNN layers=1, # of shared Bi-LSTM layers=2, hidden dimension =1000, # of attention heads=4, image level and object level feature dimension =2048, # of spatial location in image level feature =100, # of objects in object level feature=36, # of rank in bi-linear pooling=3, # of bilinear attention maps=8, # of epochs=100, initial learning rate=0.002. Optimal values of the hyperparameters are chosen based on the model performance on the development set of VQA v1.0

Dataset	Models	Overall	Other	Number	Yes/No
English	MFB	58.69	47.89	34.80	81.13
	MFH	59.07	48.04	35.42	81.73
	BUTD	63.50	54.66	38.81	83.60
	Bi-linear Attention	63.85	54.56	41.08	81.91
	Proposed Model	<b>65.37</b>	<b>56.41</b>	<b>43.84</b>	<b>84.67</b>
Hindi	MFB	57.06	46.00	33.63	79.70
	MFH	57.47	46.45	34.27	79.97
	BUTD	60.15	50.90	37.44	80.13
	Bi-linear Attention	62.50	52.99	40.31	82.66
	Proposed Model	<b>64.51</b>	<b>55.37</b>	<b>42.09</b>	<b>84.21</b>
Code-mixed	MFB	57.06	46.00	33.63	79.70
	MFH	57.10	46.09	33.56	79.71
	BUTD	60.51	51.68	36.47	80.37
	Bi-linear Attention	61.53	52.00	39.86	81.53
	Proposed Model	<b>64.69</b>	<b>55.58</b>	<b>42.57</b>	<b>84.28</b>

Table 1: Performance comparison between the state-of-the-art baselines and our proposed model on the VQA datasets. All the accuracy figures are shown in %. The improvements over the baselines are statistically significant as  $p < 0.05$  for t-test. At the time of testing, only one language input is given to the model.

dataset. Adamax optimizer (Kingma and Ba, 2014) is used to optimize the weights during training.

## 5.3 Results

In order to compare the performance of our proposed model, we define the following baselines: MFB (Yu et al., 2017), MFH (Yu et al., 2018), Bottom-up-Attention (Anderson et al., 2018) and Bi-linear Attention Network (Kim et al., 2018). These are the state-of-the-art models for VQA. We report the performance in Table 1.

The trained multilingual model is evaluated on the English VQA and MCVQA datasets as discussed in Section 5.1. Results of these experiments are reported in Table 1. Our proposed model outperforms the state-of-the-art English (with 65.37% overall accuracy), and achieves overall accuracy of 64.51% and 64.69% on Hindi and Code-mixed VQA, respectively. Due to the shared hierarchical question encoder, our proposed model learns complementary features across questions of different languages.

## 5.4 Comparison to the non-English VQA

Gao et al. (2015) created a VQA dataset for Chinese question-answer pairs and translated them to English. Their model takes the Chinese equivalent English question as input and generates an answer. A direct comparison in terms of performance is not feasible as they treat the problem as seq2seq learning (Sutskever et al., 2014) and their model was also trained on a monolingual (English) setup.

<sup>6</sup><https://github.com/libindic/indic-trans>

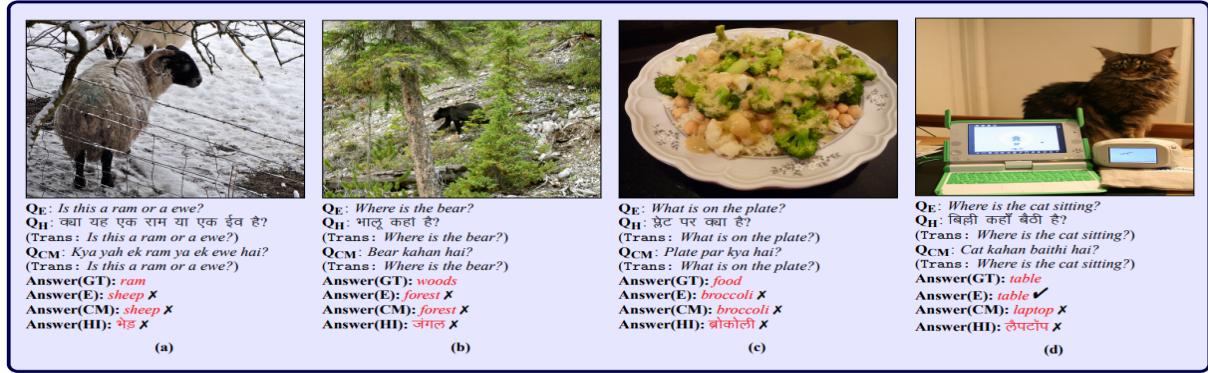


Figure 4: Some examples from MCVQA dataset where our model predicted incorrect answers. The notations are as follows:; **GT**: Ground Truth, **E**: English, **CM**: Code-mixed, **HI**: Hindi

Dataset	Training	Overall	Other	Number	Yes/No
English	English	63.85	54.56	41.08	83.91
Hindi		24.13	4.41	0.34	58.46
Code-mixed		28.04	11.95	0.49	58.79
English	Hindi	27.13	1.33	0.38	70.59
Hindi		62.50	52.99	40.31	82.66
Code-mixed		27.16	1.34	0.36	70.65
English	Code-mixed	30.92	9.45	0.39	69.85
Hindi		21.76	2.02	0.35	36.22
Code-mixed		61.53	52.00	39.86	81.53

Table 2: Results of cross-lingual experiments by training the Kim et al. (2018) model on the training dataset of one language and evaluating on the rest.

We use their question encoding and language feature interaction component to train a model with English question and achieve overall accuracy of 57.89% on English validation dataset (our model achieves 65.37%). Recently, Shimizu et al. (2018) created a dataset for Japanese question-answer pairs and applied transfer learning to predict Japanese answers from the model trained on English questions. We adopt their approach, evaluate the model on English VQA and MCVQA dataset, and achieve 61.12%, 58.23%, 58.97% overall accuracy on English, Hindi and Code-mixed, respectively. In comparison to these, we rather solve a more challenging problem that involves both multilingualism and code-mixing.

## 5.5 Analysis and Discussion

We perform ablation study to analyze the contribution of various components of our proposed system. Table 3 shows the model performance by removing one component at a time. The self-attention on question and object-level features seem to have the maximum effect on the model’s performance. The object-level features contribute more as compared to the image-level features because the object level

Models Component	English	Hindi	Code-mixed
Proposed	65.37	64.51	64.69
(-) CNN Layer	64.92	64.19	64.38
(-) Layer Attention	64.52	63.72	63.89
(-) Self Attention	64.31	63.59	63.63
(-) Image Level	64.88	63.97	64.10
(-) Object Level	64.29	63.39	63.52
(-) Counter	64.67	64.03	63.92
(-) Image (Language-only)	40.89	45.70	45.23
(-) Question (Vision-only)	24.13	26.44	21.68

Table 3: Effect of various components of the model in terms of overall accuracy on English, Hindi and Code-mixed VQA datasets. (-) X shows the VQA model architecture after removal of component ‘X’

features focus on encoding the objects of an image, which assist in answering the questions more accurately. Image grid level features help the model to encode those parts of the image which could not be encoded by the object level features.

The proposed VQA model is built on two channels: vision (image) and language (question). We perform a study (Table 3) to know the impact of both the channels on the final prediction of the model. We turn off vision features and train the model with the textual features to assess the impact of vision (image) features. Similarly, we also measure the performance of the system with image features (object and image level) only. Our study provides answer to the following question: “*How much does a VQA model look at these channels to provide an answer?*”. The study reveals that the proposed VQA model is strongly coupled with both the vision and language channels. This confirms that the outperformance of the model is not because of the textual similarity between questions or pixel-wise similarity between the images.

We also perform experiments to evaluate the system in a cross-lingual setting. Towards this, we train the best baseline system (Kim et al., 2018) on the training dataset of one language and evaluate it on test datasets of the other two. The model performs pretty well when the languages for training and validation are the same. However, the performance of the model drops significantly when it is trained on one language and evaluated on a different language. We analyze the answers predicted by the model and make following observations: (1) Our model learns the question representation from different surface forms (English, Hindi and Hinglish) of the same word. It helps for much better representation of multilingual questions by encoding their linguistic properties. These rich information also interact with the image and extract language independent joint representation of question and image. However, the state-of-the-art models are language dependent. The question representation obtained from the state-of-the-art models could not learn language independent features. Therefore, they perform poorly in cross-lingual and multilingual setups (results are reported in Table 2). (2) We observe that the model performance on English VQA dataset is slightly better than Hindi and Code-mixed. One possible reason could be that the object-level features are extracted after training on the English Visual Genome dataset. Our VQA approach is language agnostic and can be extended to other languages as well.

**Error Analysis:** We perform a thorough analysis of the errors encountered by our proposed model on English VQA and MCVQA datasets. We categorize the following major sources of errors:

(i) **Semantic similarity:** This error occurs when an image can be interpreted in two ways based on its visual surroundings. In those scenarios, our model sometimes predicts the incorrect answer that is semantically closer to the ground truth answer. For example, in Figure 4(b), the question is *Where is the bear?*. Our model predicts the *forest* as the answer. However, the ground truth answer is *woods* which is semantically similar to *forest* and is a reasonable answer.

(ii) **Ambiguity in object recognition:** This error occurs when objects of an image have similar object and image-level features. For example, in Figure 4(a) the question is *Is this a ram or a ewe?*. Our model predicts *sheep* as the answer in all the three setups, but the ground truth answer is *ram*. As a

*sheep*, a *ram* and an *ewe* have similar object and image-level features and all of them resemble the same, our model could not predict the correct answer in such cases.

(iii) **Object detection at fine-grained level:** This type of errors occur, when our model focuses on the fine-grained attributes of an image. In Figure 4(c), the question is *What is on the plate?*. The ground truth answer for this question is *food*. However, our model predicts *broccoli* as the answer. The food that is present on the plate is *broccoli*. This shows that our model is competent enough to capture the fine-grained characteristics of the image and thus predicts an incorrect answer.

(iv) **Cross-lingual training of object-level features:** Our proposed model has the capability to learn question features across multiple languages. However, the object-level features used in this work are trained on English language dataset (Visual Genome dataset). We observe (c.f. Figure 4(d)) that the model sometimes fails when the question is in Hindi or Hinglish.

## 6 Conclusion

In this work, we propose a unified end-to-end framework for multilingual and Code-mixed question answering and create a dataset for Hindi and Code-mixed VQA. We believe this dataset will enable the research in multilingual and code-mixed VQA. Our unified end-to-end model is capable of predicting answers for English, Hindi and Code-mixed questions. Experiments show that we achieve state-of-the-art performance on multilingual VQA. We believe our work will pave the way towards creation of multilingual and Code-mixed AI assistants. In the future, we plan to explore transformer-based architectures for VQA in multilingual and code-mixed setups considering various diverse languages.

## Acknowledgment

Asif Ekbal gratefully acknowledges the Young Faculty Research Fellowship (YFRF) Award supported by the Visvesvaraya PhD scheme for Electronics and IT, Ministry of Electronics and Information Technology (MeitY), Government of India, and implemented by Digital India Corporation (formerly Media Lab Asia).

## References

- Asma Ben Abacha, Soumya Gayen, Jason J Lau, Sivaraprabha Rajaraman, and Dina Demner-Fushman. 2018. Nlm at imageclef 2018 visual question answering in the medical domain. In *CLEF (Working Notes)*.
- Rodrigo Agerri, Josu Bermudez, and German Rigau. 2014. IXA Pipeline: Efficient and Ready to Use Multilingual NLP Tools. In *LREC*, pages 3823–3828.
- Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering. In *CVPR*, pages 6077–6086.
- Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016. Neural Module Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 39–48.
- Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. 2015. VQA: Visual Question Answering. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2425–2433.
- Kalika Bali, Jatin Sharma, Monojit Choudhury, and Yogarshi Vyas. 2014. "I am borrowing ya mixing?" An Analysis of English-Hindi Code Mixing in Facebook. In *Proceedings of the First Workshop on Computational Approaches to Code Switching*, pages 116–126.
- Asma Ben Abacha, Sadid A. Hasan, Vivek V. Datla, Joey Liu, Dina Demner-Fushman, and Henning Müller. 2019. VQA-Med: Overview of the medical visual question answering task at imageclef 2019. In *CLEF2019 Working Notes*, CEUR Workshop Proceedings, Lugano, Switzerland. CEUR-WS.org.
- Hedi Ben-Younes, Rémi Cadene, Matthieu Cord, and Nicolas Thome. 2017. Mutan: Multimodal Tucker Fusion for Visual Question Answering. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2612–2620.
- Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching Word Vectors with Subword Information. *arXiv preprint arXiv:1607.04606*.
- Ondrej Bojar, Vojtech Diatka, Pavel Rychlý, Pavel Stranák, Vít Suchomel, Aleš Tamchyna, and Daniel Zeman. 2014. HindEnCorp-Hindi-English and Hindi-only Corpus for Machine Translation. In *LREC*, pages 3550–3555.
- Wei-Lun Chao, Hexiang Hu, and Fei Sha. 2018. Cross-Dataset Adaptation for Visual Question Answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5716–5725.
- Xilun Chen and Claire Cardie. 2018. Unsupervised Multilingual Word Embeddings. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 261–270, Brussels, Belgium. Association for Computational Linguistics.
- Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A Large-Scale Hierarchical Image Database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255.
- Akira Fukui, Dong Huk Park, Daylen Yang, Anna Rohrbach, Trevor Darrell, and Marcus Rohrbach. 2016. Multimodal Compact Bilinear Pooling for Visual Question Answering and Visual Grounding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 457–468. Association for Computational Linguistics.
- Björn Gambäck and Amitava Das. 2014. On Measuring the Complexity of Code-Mixing. In *Proceedings of the 11th International Conference on Natural Language Processing, Goa, India*, pages 1–7.
- Haoyuan Gao, Junhua Mao, Jie Zhou, Zhiheng Huang, Lei Wang, and Wei Xu. 2015. Are You Talking to a Machine? Dataset and Methods for Multilingual Image Question Answering. In *Advances in Neural Information Processing Systems*, pages 2296–2304.
- Marcos Garcia and Pablo Gamallo. 2015. Yet Another Suite of Multilingual NLP Tools. In *International Symposium on Languages, Applications and Technologies*, pages 65–75. Springer.
- Souwick Ghosh, Satanu Ghosh, and Dipankar Das. 2017. Complexity metric for code-mixed social media text. *Computación y Sistemas*, 21(4):693–701.
- Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the V in VQA Matter: Elevating the Role of Image Understanding in Visual Question Answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6904–6913.
- Deepak Gupta, Asif Ekbal, and Pushpak Bhattacharyya. 2018a. A deep neural network based approach for entity extraction in code-mixed Indian social media text. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*, Miyazaki, Japan. European Languages Resources Association (ELRA).
- Deepak Gupta, Asif Ekbal, and Pushpak Bhattacharyya. 2019. A Deep Neural Network Framework for English Hindi Question Answering. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 19(2):1–22.
- Deepak Gupta, Pabitra Lenka, Asif Ekbal, and Pushpak Bhattacharyya. 2018b. Uncovering Code-Mixed Challenges: A Framework for Linguistically Driven

- Question Generation and Neural Based Question Answering. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 119–130. Association for Computational Linguistics.
- Deepak Gupta, Swati Suman, and Asif Ekbal. 2021. Hierarchical deep multi-modal network for medical visual question answering. *Expert Systems with Applications*, 164:113993.
- Deepak Gupta, Shubham Tripathi, Asif Ekbal, and Pushpak Bhattacharyya. 2016. A Hybrid Approach for Entity Extraction in Code-Mixed Social Media Data. *MONEY*, 25:66.
- Deepak Kumar Gupta, Shubham Kumar, and Asif Ekbal. 2014. Machine Learning Approach for Language Identification & Transliteration. In *Proceedings of the Forum for Information Retrieval Evaluation*, pages 60–64. ACM.
- Sadid A Hasan, Yuan Ling, Oladimeji Farri, Joey Liu, M Lungren, and H Müller. 2018. Overview of the ImageCLEF 2018 Medical Domain Visual Question Answering Task. In *CLEF2018 Working Notes. CEUR Workshop Proceedings, CEUR-WS, Avignon, France*.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778.
- Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural computation*, 9(8):1735–1780.
- Hexiang Hu, Wei-Lun Chao, and Fei Sha. 2018. Learning Answer Embeddings for Visual Question Answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5428–5436.
- Ronghang Hu, Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Kate Saenko. 2017. Learning to Reason: End-to-end Module Networks for Visual Question Answering. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 804–813.
- Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. 2017. CLEVR: A Diagnostic Dataset for Compositional Language and Elementary Visual Reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1988–1997.
- Kushal Kafle and Christopher Kanan. 2017. Visual Question Answering: Datasets, Algorithms, and Future Challenges. In *Computer Vision and Image Understanding*, volume 163, pages 3–20. Elsevier.
- Jin-Hwa Kim, Jaehyun Jun, and Byoung-Tak Zhang. 2018. Bilinear Attention Networks. In *Advances in Neural Information Processing Systems 31*, pages 1571–1581.
- Jin-Hwa Kim, Sang-Woo Lee, Donghyun Kwak, Min-Oh Heo, Jeonghee Kim, Jung-Woo Ha, and Byoung-Tak Zhang. 2016. Multimodal Residual Learning for Visual QA. In *Advances In Neural Information Processing Systems 29*, pages 361–369.
- Diederik P Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980*.
- Nikita Kitaev and Dan Klein. 2018. Constituency Parsing with a Self-Attentive Encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2676–2686. Association for Computational Linguistics.
- Qing Li, Qingyi Tao, Shafiq Joty, Jianfei Cai, and Jiebo Luo. 2018. VQA-E: Explaining, Elaborating, and Enhancing Your Answers for Visual Questions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 552–567.
- Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 workshop*, volume 8. Barcelona, Spain.
- Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft COCO: Common Objects in Context. In *European Conference on Computer Vision*, pages 740–755. Springer.
- Yun Liu, Xiaoming Zhang, Feiran Huang, and Zhoujun Li. 2018. Adversarial Learning of Answer-Related Representation for Visual Question Answering. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 1013–1022. ACM.
- Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. 2016. Hierarchical Question-Image Co-Attention for Visual Question Answering. In *Advances In Neural Information Processing Systems*, pages 289–297.
- Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421. Association for Computational Linguistics.
- Mateusz Malinowski and Mario Fritz. 2014. A Multi-World Approach to Question Answering about Real-World Scenes based on Uncertain Input. In *Advances in Neural Information Processing Systems*, pages 1682–1690.

- Christof Monz and Bonnie J Dorr. 2005. Iterative Translation Disambiguation for Cross-language Information Retrieval. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 520–527. ACM.
- Carol Myers-Scotton. 1997. *Duelling Languages: Grammatical Structure in Codeswitching*. Oxford University Press.
- Carol Myers-Scotton. 2002. *Contact Linguistics: Bilin-gual Encounters and Grammatical Outcomes*. Oxford University Press on Demand.
- Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. 2012. Indoor segmentation and support inference from rgbd images. In *ECCV*.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 311–318. Association for Computational Linguistics.
- Rana D. Parshad, Suman Bhowmick, Vineeta Chand, Nitu Kumari, and Neha Sinha. 2016. What is India speaking? Exploring the “Hinglish” invasion. *Physica A: Statistical Mechanics and its Applications*, 449(C):375–389.
- Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2017. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(6):1137–1149.
- Koustav Rudra, Shruti Rijhwani, Rafiya Begum, Kalika Bali, Monojit Choudhury, and Niloy Ganguly. 2016. Understanding Language Preference for Expression of Opinion and Sentiment: What do Hindi-English Speakers do on Twitter? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1131–1141.
- Kevin J Shih, Saurabh Singh, and Derek Hoiem. 2016. Where To Look: Focus Regions for Visual Question Answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4613–4621.
- Nobuyuki Shimizu, Na Rong, and Takashi Miyazaki. 2018. Visual Question Answering Dataset for Bilingual Image Understanding: A Study of Cross-Lingual Transfer Using Attention Maps. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1918–1928.
- Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of association for machine translation in the Americas*, volume 200.
- Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to Sequence Learning with Neural Networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.
- Peng Wang, Qi Wu, Chunhua Shen, Anthony Dick, and Anton van den Hengel. 2017. Explicit Knowledge-based Reasoning for Visual Question Answering. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 1290–1296.
- Peng Wang, Qi Wu, Chunhua Shen, Anthony Dick, and Anton van den Hengel. 2018. FVQA: Fact-based Visual Question Answering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40:2413–2427.
- Qi Wu, Peng Wang, Chunhua Shen, Ian Reid, and Anton van den Hengel. 2018. Are You Talking to Me? Reasoned Visual Dialog Generation through Adversarial Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6106–6115.
- Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google’s Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *CoRR*, abs/1609.08144.
- Huijuan Xu and Kate Saenko. 2016. Ask, Attend and Answer: Exploring Question-Guided Spatial Attention for Visual Question Answering. In *European Conference on Computer Vision*, pages 451–466. Springer.
- Zhou Yu, Jun Yu, Jianping Fan, and Dacheng Tao. 2017. Multi-Modal Factorized Bilinear Pooling With Co-Attention Learning for Visual Question Answering. In *Proceedings of the IEEE International Conference on Computer Vision*.
- Zhou Yu, Jun Yu, Chenchao Xiang, Jianping Fan, and Dacheng Tao. 2018. Beyond Bilinear: Generalized Multi-modal Factorized High-order Pooling for Visual Question Answering. *IEEE Transactions on Neural Networks and Learning Systems*, (99):1–13.
- Joe Yue-Hei Ng, Matthew Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and

George Toderici. 2015. Beyond Short Snippets: Deep Networks for Video Classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4694–4702.

Yan Zhang, Jonathon Hare, and Adam Prügel-Bennett. 2018. Learning to Count Objects in Natural Images for Visual Question Answering. In *International Conference on Learning Representations*.

Zhou Zhao, Qifan Yang, Deng Cai, Xiaofei He, and Yueteng Zhuang. 2017. Video Question Answering via Hierarchical Spatio-Temporal Attention Networks. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3518–3524.

Yangyang Zhou, Xin Kang, and Fuji Ren. 2018. Employing inception-resnet-v2 and bi-lstm for medical domain visual question answering. In *CLEF (Working Notes)*.

Yuke Zhu, Oliver Groth, Michael Bernstein, and Li Fei-Fei. 2016. Visual7w: Grounded Question Answering in Images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4995–5004.

## A Appendices

The detailed comparison of automatically created and manually code-mixed questions w.r.t the Code-mixing Index (CMI) score, Complexity Factor (CF2 and CF3) are shown in Table 4. We also show the comparison of our MCVQA dataset with other VQA datasets in Table 5. The analysis of MCVQA dataset are illustrated in Fig 5 and 6.

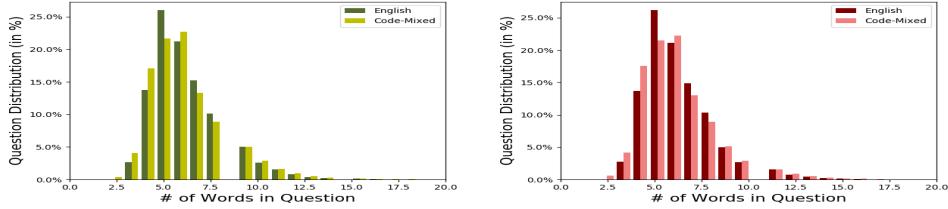


Figure 5: Analysis of question distribution w.r.t the question length between VQA v1.0 English and code-mixed, train and test dataset.

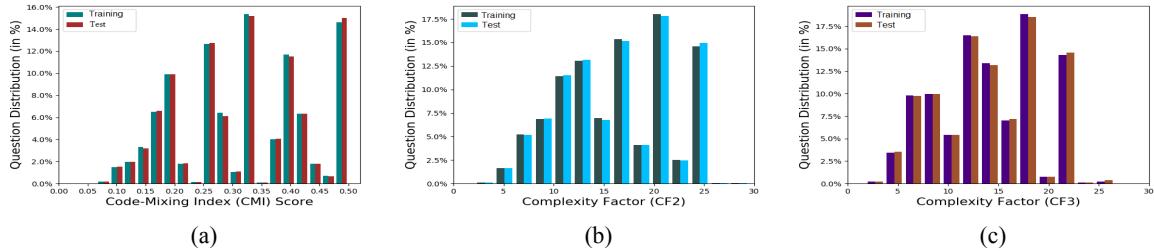


Figure 6: Analysis of code-mixed VQA dataset on various code-mixing metrics: (a), (b) and (c) show the distribution of code-mixed questions from training and test set w.r.t the Code-mixing Index (CMI) score, Complexity Factor (CF2 and CF3), respectively.

Metrics	Manually Annotated	Automatically Generated	
		Training	Testing
CMI Score	0.2946	0.3223	0.3228
CF2	14.765	16.094	16.114
CF3	13.122	14.0708	14.096

Table 4: Comparison between manually annotated code-mixed questions and automatically generated code-mixed questions w.r.t the CMI score, CF2, and CF3.

Dataset	Images used	Created by	Multilingual	Code-Mixed
DAQUAR (Malinowski and Fritz, 2014)	NYU Depth V2	In-house participants, Automatically generated	✗	✗
FM-IQA (Gao et al., 2015)	MSCOCO	Crowd workers (Baidu)	✓	✗
VQA v1.0 (Antol et al., 2015)	MSCOCO	Crowd workers (AMT)	✗	✗
Visual7W (Zhu et al., 2016)	MSCOCO	Crowd workers (AMT)	✗	✗
CLEVR (Johnson et al., 2017)	Synthetic Shapes	Automatically generated	✗	✗
KB-VQA (Wang et al., 2017)	MSCOCO	In-house participants	✗	✗
FVQA (Wang et al., 2018)	MSCOCO	In-house participants	✗	✗
Japanese VQA (Shimizu et al., 2018)	MSCOCO	Crowd workers (Yahoo)	✓	✗
<b>MCVQA (Ours)</b>	<b>MSCOCO</b>	<b>Automatically generated</b>	<b>✓</b>	<b>✓</b>

Table 5: Comparison of VQA datasets with our MCVQA dataset. The images used are: MSCOCO (Lin et al., 2014) and NYU Depth v2 (Nathan Silberman and Fergus, 2012)

# Toxic Language Detection in Social Media for Brazilian Portuguese: New Dataset and Multilingual Analysis

João A. Leite, Diego F. Silva

Departamento de Computação  
Federal University of São Carlos  
São Carlos, Brazil

joaoaugustobr@hotmail.com  
diegofs@ufscar.br

Kalina Bontcheva, Carolina Scarton

Department of Computer Science  
University of Sheffield  
Sheffield, UK

k.bontcheva@sheffield.ac.uk  
c.scarton@sheffield.ac.uk

## Abstract

Hate speech and toxic comments are a common concern of social media platform users. Although these comments are, fortunately, the minority in these platforms, they are still capable of causing harm. Therefore, identifying these comments is an important task for studying and preventing the proliferation of toxicity in social media. Previous work in automatically detecting toxic comments focus mainly in English, with very few work in languages like Brazilian Portuguese. In this paper, we propose a new large-scale dataset for Brazilian Portuguese with tweets annotated as either toxic or non-toxic or in different types of toxicity. We present our dataset collection and annotation process, where we aimed to select candidates covering multiple demographic groups. State-of-the-art BERT models were able to achieve 76% macro- $F_1$  score using monolingual data in the binary case. We also show that large-scale monolingual data is still needed to create more accurate models, despite recent advances in multilingual approaches. An error analysis and experiments with multi-label classification show the difficulty of classifying certain types of toxic comments that appear less frequently in our data and highlights the need to develop models that are aware of different categories of toxicity.

## 1 Introduction

Social media can be a powerful tool that enables virtual human interactions, connecting people and enhancing businesses' presence. On the other hand, since users feel somehow protected under their virtual identities, social media has also become a platform for hate speech and use of toxic language. Although hate speech is a crime in most countries, identifying cases in social media is not an easy task, given the massive amounts of data posted every day. Therefore, automatic approaches for detecting online hate speech have received significant attention

in recent years (Waseem and Hovy, 2016; Davidson et al., 2017; Zampieri et al., 2019b). In this paper, we focus on the analysis and automatic detection of **toxic comments**. Our definition of toxic is similar to the one used by the Jigsaw competition,<sup>1</sup> where comments containing insults and obscene language are also considered, besides hate speech.<sup>2</sup> Systems capable of automatically identifying toxic comments are useful for platform's moderators and to select content for specific users (e.g. children). Nevertheless, there are multiple challenges specific to process toxic comments automatically, e.g. (i) toxic language may not be explicit, i.e. may not contain explicit toxic terms; (ii) there is a large spectrum of types of toxicity (e.g. sexism, racism, insult); (iii) toxic comments correspond to a minority of comments, which is fortunate, but means that automatic data-driven approaches need to deal with highly unbalanced data.

Although there is some work on this topic for other languages – e.g. Arabic (Mubarak et al., 2017) and German (Wiegand et al., 2018) –, most of the resources and studies available are for English (Davidson et al., 2017; Wulczyn et al., 2017; Founta et al., 2018; Mandl et al., 2019; Zampieri et al., 2019b).<sup>3</sup> For Portuguese, only two previous works are available (Fortuna et al., 2019; de Pelle and Moreira, 2017) and their datasets are considerably small, mainly when compared to resources available for English.

We present **ToLD-Br** (Toxic Language Dataset for Brazilian Portuguese), a new dataset with Twitter posts in the Brazilian Portuguese language.<sup>4</sup>

<sup>1</sup><https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge/overview>

<sup>2</sup>This is also similar to the usage of *offensive* comments in OffensEval (Zampieri et al., 2019b, 2020).

<sup>3</sup>A large list of resources is available at <http://hatespeechdata.com>.

<sup>4</sup>It is important to distinguish the language variant, since

A total of 21K tweets were manually annotated into seven categories: *non-toxic*, *LGBTQ+phobia*, *obscene*, *insult*, *racism*, *misogyny* and *xenophobia*. Each tweet has three annotations that were made by volunteers from a university in Brazil. Volunteers were selected taking into account demographic information, aiming to create a dataset as balanced as possible in regarding to demographic group biases. This is then the largest dataset available for toxic data analysis in social media for the Portuguese language and the first dataset with demographic information about annotators.<sup>5</sup>

We experiment with Brazilian Portuguese (Souza et al., 2019) and Multilingual (Wolf et al., 2019) BERT models (Devlin et al., 2019) for the binary task of **automatically classifying toxic comments**, since similar models achieve state-of-the-art results for the same task in other languages (Zampieri et al., 2019b). Models fine-tuned on monolingual data achieve up to 76% of macro-*F1*, improving 3 points over a baseline. Besides, BERT-based approaches with multilingual pre-trained models enable transfer learning and zero-shot learning. The OffensEval 2019 OLID dataset (Zampieri et al., 2019a) is then used to experiment with (i) **transfer-learning**: where both OLID and ToLD-Br are used to fine-tune BERT; and, (ii) **zero-shot learning**: where BERT is fine-tuned using only OLID. Results highlight the importance of language-specific datasets, since transfer learning does not improve over monolingual models and zero-shot learning achieves only a macro-*F1* of 56%.

**An error analysis** is performed using our best model, where the worst-case scenario, i.e., classifying *toxic* comments as *non-toxic*, is further investigated, taking into account the fine-grained categories. Results show that categories with fewer examples in the dataset (*racism* and *xenophobia*) are more likely to be mislabelled than other classes, with the best performance being achieved by majority classes (*insult* and *obscene*). We also analyse the **amount of data** needed in order to achieve the best performance in binary classification. Models trained with few examples are only accurate in predicting the majority class (*non-toxic*). As the number of instances grow, the performance on the minority class (*toxic*) improves significantly.

---

there are multiple differences between Brazilian Portuguese lexicon and other variants of Portuguese.

<sup>5</sup>ToLD-Br is available at: <https://github.com/JAugusto97/ToLD-Br>

Finally, we experiment with **multi-label classification**, where each different type of toxicity is automatically classified. This is a considerably harder problem than binary classification, where BERT-based models do not outperform the baseline.

Section 2 presents an overview of relevant previous work. Section 3 shows details about the ToLD-Br dataset. Material and methods are presented in Section 4, whilst results are discussed in Section 5. Finally, Section 6 shows a final discussion and future work.

## 2 Related Work

Although multiple researchers have addressed the topic of hate speech (e.g. Waseem and Hovy (2016), Chung et al. (2019), Basile et al. (2019)), we focus the literature review on previous work related to toxic comments detection, the topic of our paper. Due to space constraints, we only describe papers that create and use Twitter-based datasets and/or focus on the Brazilian Portuguese language.

**English** Davidson et al. (2017) present a dataset with around 25K tweets annotated by crowd-workers as containing *hate*, *offensive language*, or *neither*. They build a feature-based classifier with TF-IDF transformation over *n*-grams, part-of-speech information, sentiment analysis, network information (e.g., number of replies), among other features. Their best model, trained using logistic regression, achieves a macro-*F1* of 90. Founta et al. (2018) also rely on crowd-workers to annotate 80K tweets into eight categories: *offensive*, *abusive*, *hateful speech*, *aggressive*, *cyberbullying*, *spam*, and *normal*. They perform an exploratory approach to identify the categories that cause most confusion to crowd-workers. Their final, large-scale annotation is done using four categories: *abusive*, *hateful*, *normal*, or *spam*. OffensEval is a series of shared tasks focusing on offensive comments detection (Zampieri et al., 2019b, 2020). The OLID dataset (used in the 2019 edition) has around 14K tweets in English manually annotated as *offensive* or *non-offensive*. The best model for the relevant task A (*offensive* versus *non-offensive*) uses a BERT-based classifier and achieves 82.9 of macro-*F1*.

**German** A shared task (organized as part of GermEval 2018) aimed to classify tweets in German categorized into *offensive* or *non-offensive* (Wiegand et al., 2018). They make available a manually annotated dataset with approximately 8.5K tweets.

The best system achieved 76.77 of  $F_1$ -score and was a feature-based ensemble approach.

**Arabic** Mubarak et al. (2017) present a dataset with 1.1K manually annotated tweets into *obscene*, *offensive*, or *clean*. They experiment with lexical-based approaches that achieve a maximum of 60  $F_1$ -score. Mulki et al. (2019) create a dataset with tweets in the Levantine dialect of Arabic manually annotated into *normal*, *abusive*, or *hate* (with approximately 5K tweets). The authors use feature-based approaches to induce models for ternary and binary scenarios, with best systems achieving 74.4 and 89.6 of  $F_1$ -score, respectively.

**Spanish** Carmona et al. (2018) present a shared task aiming to detect aggressive tweets in Mexican Spanish. They manually annotate 11K tweets into *aggressive* or *non-aggressive*. The best system is a feature-based approach with macro- $F_1$  of 62.

**Hindi** Mathur et al. (2018) present a dataset of around 3.6K tweets in Hinglish (spoken Hindi written using the Roman script). The dataset was annotated into three classes *not offensive*, *abusive* and *hate-inducing* by ten NLP researchers. A Convolutional Neural Network (CNN) architecture with transfer learning is used, where the model is trained with both Hinglish and English data (from (Davidson et al., 2017)), achieving 71.4% of  $F_1$ -score.

**Portuguese** de Pelle and Moreira (2017) make available a dataset with 1,250 comments, extracted from comment sessions of [g1.globo.com](http://g1.globo.com) website, and annotated them into categories of *offensive* or *non-offensive*. The offensive class was also subdivided into *racism*, *sexism*, *LGBTQ+phobia*, *xenophobia*, *religious in-tolerance*, or *cursing*. They experiment with binary classification, using  $n$ -grams as features to SVM and NaiveBayes models. Best results are achieved with SVM reaching a weighted F1 score between 77 and 82, depending on different label interpretations. Fortuna et al. (2019) describe a dataset with 5,668 tweets classified as *hate* vs. *non-hate*, with the *hate* class further classified following a fine-grained hierarchy. Experiments with binary classification show a  $F_1$  score of 78 using an LSTM-based architecture.

**Multilingual** HASOC was a shared task aiming to classify hate speech and offensive comments in English, German, and Hindi (Mandl et al., 2019). Their dataset contains around 7K tweets and Facebook posts manually annotated. Sub-task A sep-

arates posts into *hate speech* or *offensive* versus *neither*; and, sub-task B separates posts containing *hate speech* or *offence* into three categories: *hate speech*, *offensive* or *profane*. Best performing systems in all languages used deep learning approaches. For OffensEval 2020 (Zampieri et al., 2020), a more extensive training data is available for English (over 9M tweets), although the annotation was made semi-automatically. Arabic, Danish, Greek, and Turkish datasets are also available with manually annotated labels. For all languages, best models are achieved using some variation of BERT.

Our work is different from previous approaches because we (i) release a large-scale dataset for a language other than English, that was created with the aim to reduce demographic biases; (ii) experiment with multilingual approaches, including transfer learning and zero-shot-learning; (iii) perform an analysis of the amount of data needed to train reliable models; and, (iv) experiment with multi-label classification, providing first insights into this challenge task.

### 3 Dataset

In this section, we describe the procedure adopted to create ToLD-Br and present its main features.

#### 3.1 Data collection

We used the GATE Cloud’s Twitter Collector<sup>6</sup> to collect posts on the Twitter platform from July to August 2019. We used two different strategies to select tweets for ToLD-Br, aiming to increase the probability of obtaining posts with toxic content, given that the volume of toxic tweets is significantly smaller than data without offensive language. Our first strategy searches for tweets that mention predefined hashtags or keywords. We chose predefined terms highly likely to belong to a toxic tweet in Brazilian Twitter, such as *gay* (“*Gay tem que apanhar*” – “*Gay should be beaten up*”), *mulherzinha* (“*Mulherzinha, vai lavar louça*” – “*Sissy, go wash dishes*”), and *nordestino* (“*Nordestino preguiçoso*” – “*Lazy Northeastern*”). However, using this strategy alone may hinder learning a model capable of generalising the concept of toxicity beyond the scope of keywords. Consequently, another strategy was adopted: we scraped tweets that mention influential users like Brazil’s president Jair Bolsonaro and soccer player Neymar Jr,

<sup>6</sup><https://cloud.gate.ac.uk/shopfront/displayItem/twitter-collector>

prone to receive abuse (around 50 influential users were monitored). Tweets collected through this method have no restrictions in terms of keywords and should broaden the scope of the data.

We collected more than 10M unique tweets and randomly selected 21K examples to compose the annotated corpus. We note that 12,600 of these posts (60%) comes from the first strategy – predefined keywords – and the remaining are tweets from threads of predefined users. The data was pseudoanonymised before being sent for annotation, with all @ mentions replaced by @user.

### 3.2 Corpus annotation

The annotation process started by choosing volunteers to perform the task of assigning labels for each example. For this, we made a public consultation at the Federal University of São Carlos (Brazil) to find candidate annotators (129 volunteers registered for the task). From these candidates, 42 were selected based on their demographic information, aiming to balance annotation bias as the interpretation of toxicity may vary. Each annotator labelled 1,500 tweets, selecting one of the following categories: *LGBTQ+phobia*, *obscene*, *insult*, *racism*, *misogyny* and/or *xenophobia* (or leaving it blank for *none*). Each tweet was annotated by three different annotators.

To evaluate the diversity among the annotators, we explore their profile. We emphasise that the identity of all annotators has been preserved. At this stage, we only survey general aspects of the volunteers who joined the labelling process. Table 1 presents the distribution of annotators regarding sex, sexual orientation, and ethnicity. To define these categories, we use the same values as the Brazilian Institute of Geography and Statistics,<sup>7</sup> in addition to giving the candidate the option of not declaring a value for each characteristic. Although we tried to keep the demographic aspects as balanced as possible when selecting the annotators, our pool of volunteers was still biased towards people identified as *white* and *heterosexual* (sex is a more balanced aspect than the others). The age of the annotators varies between 18 and 37 years, with most of them in the range between 19 and 23. Figure 1 illustrates the age distribution.

We perform different data analysis over the dataset to better understand its properties. Inter-

	Categories	# annotators
Sex	Male	18
	Female	24
Sexual orientation	Heterosexual	22
	Bisexual	12
	Homosexual	5
	Pansexual	3
Ethnicity	White	25
	Brown	9
	Black	5
	Asian	2
	Non-Declared	1

Table 1: Annotators demographic information.

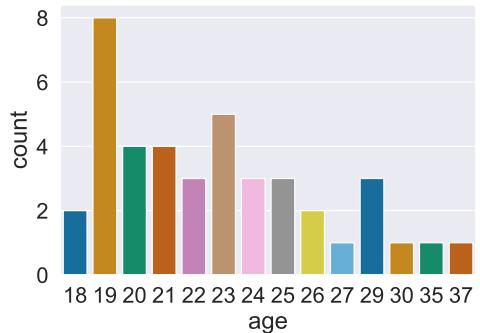


Figure 1: Annotators age distribution.

	$\alpha$
LGBTQ+phobia	0.68
Insult	0.56
Xenophobia	0.57
Misogyny	0.52
Obscene	0.49
Racism	0.48
Mean	0.55

Table 2: Krippendorff's  $\alpha$  for each label.

annotator agreement is calculated in terms of *Krippendorff's  $\alpha$*  (Table 2), since  $\alpha$  is robust to multiple annotators, different degrees of disagreement and, missing values (Artstein and Poesio, 2008).

The *LGBTQ+phobia* class shows the highest agreement, which may indicate that comments in this class have a more distinctive lexicon than other classes. The lowest agreement is seen in *obscene* and *racism* classes. Besides, we observed in the annotations many cases in which some examples were labelled as separate classes, although they intend

<sup>7</sup><https://www.ibge.gov.br/en/home-eng.html>

		Ann 1	Ann 2	Ann 3
<i>o fdp do filho dela nao parava de tocar auto pra c*****o [...]</i>		Insult	None	Obscene
<i>her sob son did not stop to play loud as f***k [...]</i>				
<i>[...] VAI SE F***R IRMÃO VC NÃO É FELIZ PQ NAO QUER</i>		Obscene	Insult	Insult
<i>[...] f***k you brother you are not happy because you do not want to be</i>				
<i>“Aonde tem um monte que fala mal, mas ninguém vai embora do morro.”</i>				
<i>acha que alguém mora aqui por que quer, c*****o!? Que idéia. [...]</i>		Obscene	Obscene	Insult
<i>“Where there are loads saying bad things, but nobody leaves the slum.”</i>				
<i>who thinks that someone lives here because they want, f***k!? What an idea. [...]</i>				

Table 3: Example of annotation divergence.

LGBTQ+phobia	Obscene	Insult	Racism	Misogyny	Xenophobia
viado (59)	porra (332)	puta (221)	nego (6)	putinha (38)	sulista (12)
boiola (15)	caralho (317)	caralho (150)	branco (6)	puta (22)	carioca (7)
viadinho (13)	puta (268)	cara (135)	preto (4)	piranha (19)	fala (4)
sapatão (12)	tomar (136)	porra (122)	nada (4)	mulher (11)	paulista (4)
caralho (11)	fuder (98)	lixo (101)	negão (3)	vagabunda (11)	gente (3)
cara (10)	cara (94)	filho (92)	cara (3)	quer (8)	nordestino (3)
quer (9)	merda (90)	burro (87)	falando (3)	vaca (8)	todo (3)
homem (9)	mano (87)	tomar (86)	vida (3)	fica (6)	ainda (3)
todo (9)	toma (85)	merda (78)	segue (2)	onde (5)	sendo (2)
bicha (9)	fazer (77)	idiota (76)	página (2)	tudo (5)	dança (2)

Table 4: The most common words of each class and the number of sentences they occur (within parentheses).

to point the same concept. Classes like *obscene* and *insult* seem to have confused the annotators, which may indicate an intersection in these concepts. Table 3 shows examples of disagreements in the classification of *obscene* and *insult*.

Table 4 presents the ten most frequent words for each class, after removing stopwords. It confirms the intersection between classes *obscene* and *insult*, with six out of ten words in common. For a quantitative analysis, Table 5 presents the *Jaccard* distance between the 100 most frequent words for each class. *Obscene* and *insult* show a considerably lower distance than other pairs (0.57), indicating that they have more words in common.

### 3.3 Dataset characteristics

For the purpose of training models for automatically classifying toxic comments, we must create aggregated annotations to provide only one binary label for each class. Different rules can be employed to aggregate the annotations, with different semantics. When we set an example as positive for toxicity only when all the annotators consider it to have the same category of offence, we insert bias to

	a	b	c	d	e	f
a	0.00	0.73	0.78	0.90	0.80	0.94
b	-	0.00	0.57	0.84	0.77	0.90
c	-	-	0.00	0.86	0.75	0.92
d	-	-	-	0.00	0.87	0.95
e	-	-	-	-	0.00	0.94

Table 5: *Jaccard* distance between all pair of classes.  
(a) LGBTQ+phobia; (b) Obscene; (c) Insult; (d) Racism; (e) Misogyny; (f) Xenophobia.

the model to not accuse a comment as toxic unless the offence is evident. Since this is very restrictive, we can also use the majority rule, but there must still be a consensus among the annotators. A last option is to consider that only a positive annotation is sufficient to label the example as positive. This procedure acknowledges that annotators may have divergent views about what was said. It is a risky rule if we intend to create rigid systems that classify the tweets and take corrective or prohibitive actions. However, it is beneficial for training a model that “raises a flag” to help moderators to assess the com-

	LGBTQ+phobia	Insult	Xenophobia	Misogyny	Obscene	Racism		Toxic
<b>At least one annotator</b>								
0	20656	16615	20849	20537	14348	20862		11745
1	344	4385	151	463	6652	138		9255
<b>At least two annotators</b>								
0	20824	19131	20958	20867	18597	20967		16566
1	176	1869	42	133	2403	33		4424
<b>Three annotators</b>								
0	20926	20483	20985	20971	20388	20994		19510
1	74	517	15	29	612	6		1490

Table 6: Dataset distribution considering different types of label aggregation.

ments. Table 6 shows the data distribution for each label and each aggregation strategy.

For the sake of reproducibility and further usage, ToLD-Br is split into default training (80%), development (10%) and test (10%) sets using a stratified strategy. Besides, the corpus is released with all the annotations. Thus, future users of ToLD-Br will be able to use it with all the labels and with varying levels of agreement between the annotators. In this paper, we consider the least restrictive case, where if at least one annotator marked any offence category in an example, the example is positive for toxicity. Likewise, if a tweet was not tagged in any of these categories, it is considered non-toxic. We believe that it is essential that if any person feels uncomfortable with a post, it should be flagged as having a certain degree of toxicity. Therefore, a model built with this data must be able to identify offensive posts, even for a specific group of people.

## 4 Materials and Methods

In this section, we describe the techniques, tools, and other materials used in our experimental evaluation. As mentioned before, we restrict our experiments on the dataset labelled as positive when at least one annotator considers the example as toxic. We then investigate the effects of the number of instances in the training data, different algorithms to train a classification model, various scenarios considering single- and multilingual models, and perform an initial experiment with multi-label classification.

We use Bag-of-Words (BoW) to represent the examples and an AutoML model to build the baseline model (BoW+AutoML). For this, we

use the `auto-sklearn`<sup>8</sup> library (Feurer et al., 2019). For our BERT-based models, we use the `simpletransformers`<sup>9</sup> library, that allows easy training and evaluation. We use default arguments for parameter tuning and define a seed to allow for reproducibility. Two versions of pre-trained BERT language models are applied: Brazilian Portuguese BERT<sup>10</sup> (Souza et al., 2019), and Multilingual BERT<sup>11</sup> (Wolf et al., 2019).

ToLD-Br is used to fine-tune BERT-based models for our monolingual experiments, with monolingual BERT (BR-BERT) and multilingual BERT (M-BERT-BR). Although M-BERT-BR refers to the multilingual version of BERT, we refer to these two models as “monolingual models,” as we trained using the dataset with Brazilian Portuguese sentences alone.

Using the multilingual model, we also carry out experiments in which we add data in English to train the models either through transfer learning or zero-shot learning. For these experiments we use the OLID data, concatenating the training and test splits into a single dataset. For transfer learning, we merged OLID and ToLD-Br to obtain a model with both languages as input, aiming to assess whether extra data in English helps in building better models (M-BERT (transfer)). For zero-shot learning, OLID is used alone at training time, building a model that did not have access to any data in Brazilian Portuguese (M-BERT (zero-shot)).

<sup>8</sup><https://automl.github.io/auto-sklearn>

<sup>9</sup>[github.com/ThilinaRajapakse/simpletransformers](https://github.com/ThilinaRajapakse/simpletransformers)

<sup>10</sup>[huggingface.co/neuralmind/bert-base-portuguese-cased](https://huggingface.co/neuralmind/bert-base-portuguese-cased)

<sup>11</sup>[huggingface.co/bert-base-multilingual-cased](https://huggingface.co/bert-base-multilingual-cased)

Through these experiments, we can assess the advantages of monolingual models, whether data from another language can directly benefit the classification, and whether a specific monolingual dataset is necessary or not.

We experiment with different sizes of the training set to assess the influence of the volume of data on the classification. For that, we evaluate the results on random subsets of the data. The size of each partition varies in a range between 10% and 100% adding 10% of the data at each iteration. For each step, we repeat the classification three times to minimise the probability of reporting results obtained by chance. Our best model (M-BERT-BR) is used for this experiment (c.f. Section 5).

Evaluation for binary classification is done in terms of precision, recall and, *F1*-score per class and macro-*F1*. We also analyse the confusion matrices of our systems in order to better visualise the performance of our models in each class, mainly focusing on an analysis of false negatives.

Although we mainly focus on binary classification, an initial approach for multi-label classification is also presented. We use the adaptation for the multi-label classification scenario available in `simpletransformers`. In this case, the transformer’s output consists of six neurons, each representing one of the labels. These neurons are considered independent in the training and prediction process. Thus, when an output neuron is activated, we set the label represented by this neuron to positive. Besides, we evaluate the performance of a baseline based on BoW+AutoML, where we train an AutoML model for multilabel classification. Evaluation is done in terms of *Hamming* loss and average precision (Tsoumakas et al., 2009).

## 5 Results and Discussion

This section shows the results of our experiments in classifying toxic comments using ToLD-Br.

### 5.1 Binary Classification

For evaluating our models, we are particularly interested in models with high performance in the positive class (classification of *toxic* comments). The worst case scenario are false negatives, i.e. *toxic* comments classified as *non-toxic*. Tables 7 through 11 summarises the results for each model. BoW+AutoML is already a competitive model, achieving 74% of macro-*F1*, as shown in Table 7 and Figure 2a.

	Precision	Recall	F1-score
0	0.76	0.75	0.75
1	0.71	0.73	0.72
Macro Avg	0.74	0.74	0.74
Weighted Avg	0.74	0.74	0.74

Table 7: BoW + AutoML

	Precision	Recall	F1-score
0	0.77	0.80	0.79
1	0.76	0.73	0.74
Macro Avg	0.76	0.76	0.76
Weighted Avg	0.76	0.77	0.76

Table 8: BR-BERT

	Precision	Recall	F1-score
0	0.81	0.69	0.75
1	0.69	0.82	0.75
Macro Avg	0.75	0.75	0.75
Weighted Avg	0.76	0.75	0.75

Table 9: M-BERT-BR

	Precision	Recall	F1-score
0	0.80	0.74	0.77
1	0.72	0.79	0.75
Macro Avg	0.76	0.76	0.76
Weighted Avg	0.77	0.76	0.76

Table 10: M-BERT(transfer)

	Precision	Recall	F1-score
0	0.59	0.83	0.69
1	0.63	0.32	0.43
Macro Avg	0.61	0.58	0.56
Weighted Avg	0.61	0.60	0.57

Table 11: M-BERT(zero-shot)

The monolingual models BR-BERT and M-BERT-BR (Tables 8 and 9, respectively) show very similar performances in all metrics, with BR-BERT being slightly better in terms of macro-*F1*. However, M-BERT-BR is better in terms of *F1*-score for the positive class and shows fewer false negatives than BR-BERT (Figure 2b for BR-BERT and Figure 2c for M-BERT-BR).

M-BERT (transfer) (Table 10) does not out-

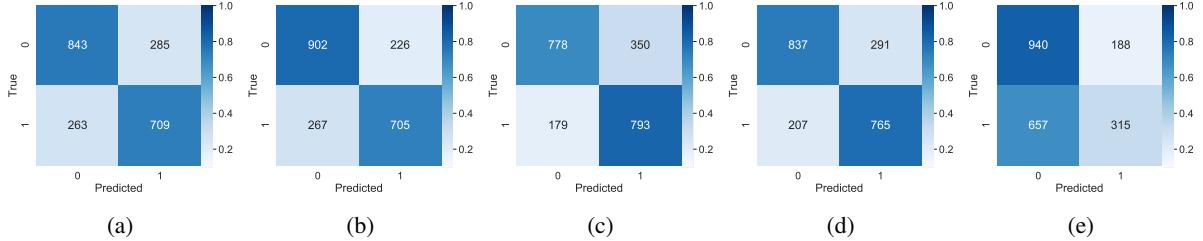


Figure 2: Confusion matrices for each model (a) BoW+AutoML (Baseline); (b) BR-BERT; (c) M-BERT-BR; (d) M-BERT(transfer); (e) M-BERT(zero-shot)

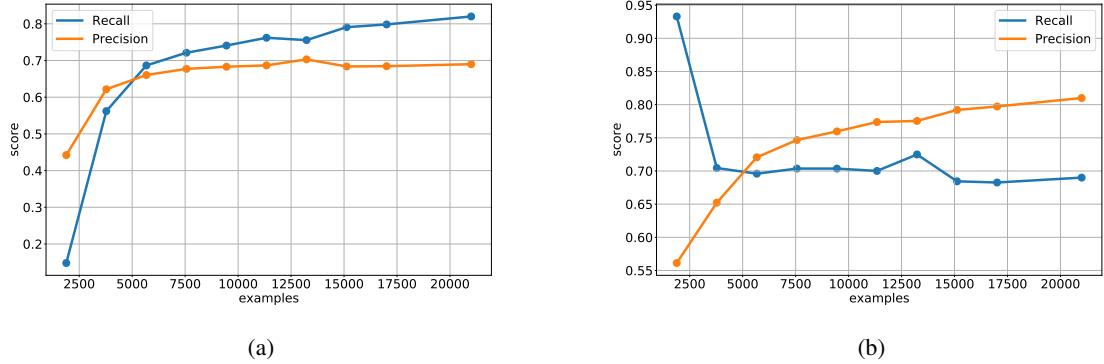


Figure 3: Precision and recall for different sizes of the training dataset for the (a) *positive* and (b) *negative* classes.

perform the monolingual models and it also shows more false negatives than M-BERT-BR (Figure 2e). On the other hand, the number of false negatives in BR-BERT (267) is slightly higher than the number of false negatives in M-BERT (transfer) (207). Finally, M-BERT (zero-shot) (Table 11) is the worst model, as expected. It performs particularly bad when classifying the positive class, achieving only 43% of  $F1$ -score for this class, mainly caused by its high number of false negatives (Figure 2d).

In summary, transfer learning does not seem to improve over the overall performance of monolingual models. Based on the analysis of false negatives, M-BERT-BR appears as our best model. Zero-shot learning shows a very low performance, being particularly bad in the positive class.

**Error Analysis** We also analyse the performance of our best model (M-BERT-BR) in each fine-grained class. The idea is to identify which toxic classes are most difficult to be classified as *toxic* by our binary classifier. As false negatives are a critical type of error in our application, Table 12 shows the false negative rate (false negatives / expected positives) for each toxic class. The ratio of false negatives is inversely proportional to the number of examples for a specific class. *Insult* and *obscene*, the largest classes, show the lowest false

negative rate, whilst the highest rates are shown by classes with less examples (*racism* and *xenophobia*). Therefore, in order to improve classification models, these aspects of the imbalanced data need to be taken into account and further studied.

	False negative rate
LGBTQ+phobia	7/35 (0.2)
Insult	67/448 (0.15)
Xenophobia	13/19 (0.68)
Misogyny	7/45 (0.15)
Obscene	117/701 (0.17)
Racism	8/17 (0.47)

Table 12: Error analysis for each label.

## 5.2 Importance of Large Datasets

In this experiment, we highlight the importance of collecting a considerable amount of examples, as toxicity can be expressed in many different ways. We separated the training data into 10 random splits from 10% to 100% of the data, increasing 10% of data at each step, and trained M-BERT-BR with three random samples for each step. Figure 3 shows the mean recall, precision and  $F1$ -score for the positive and negative classes, respectively, for each

data split. With few training examples, the model only performs well on the majority class, but as the number of instances grows, recall for the negative class starts decreasing while recall for the positive class increases, and precision rises for both classes. At least 6K examples seems to be necessary to achieve reliable results, while previous work for Portuguese reports the largest dataset with only 5,668 examples. This highlights the importance of ToLD-Br, as a large-scale dataset.

### 5.3 Multi-Label Classification

We experiment with multi-label classification, building a model using the Multilingual BERT (similar to M-BERT-BR). Our baseline is a set of BoW+AutoML models trained using Binary Relevance (Tsoumakas et al., 2009) for multi-label classification. The BERT-based models adopt a score threshold of 0.5 in the output neuron to deal with multi-label. If the activation for a label in the output layer is higher than the threshold, we consider it positive.

The baseline model obtained 0.08 and 0.20 of *Hamming* loss and average precision, respectively, while M-BERT-BR resulted in 0.07 and 0.19 for these measures, respectively. Figure 4 displays the confusion matrices obtained by M-BERT-BR.

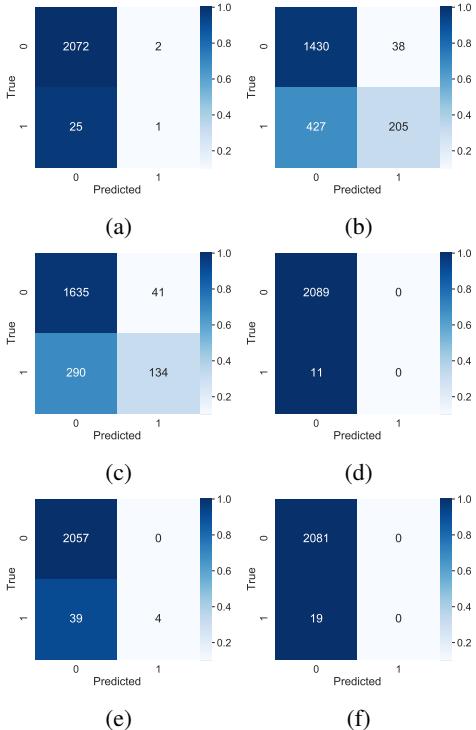


Figure 4: Confusion matrices for each label (a) *LGBTQ+phobia*; (b) *Obscene*; (c) *Insult*; (d) *Racism*; (e) *Misogyny*; (f) *Xenophobia*.

This scenario is considerably more challenging than binary classification. The positive class of each label corresponds to a subset of the examples labelled as *toxic*. Thus, it is likely that the number of instances for these classes will be insufficient for the model to learn. Besides, the problem of unbalanced classes becomes evident (c.f. Table 6). As a consequence, it is clear that labels with a small number of positive examples, like *racism*, *misogyny*, *xenophobia*, and *LGBTQ+phobia* were almost entirely classified as negative. In contrast, for *obscene* and *insult*, labels with a considerable amount of positive examples, the model was capable of classifying some examples correctly. In all cases, besides *insult*, the baseline performs slightly better for the positive class (which justify the higher *Hamming* loss). This setback is likely due to the difficulty of the neural model to learn with few examples.

## 6 Concluding Remarks

In this paper, we present ToLD-Br: a dataset for the classification of toxic comments on Twitter in Brazilian Portuguese. Through a wide and comprehensive analysis, we demonstrated the need for this dataset for studies on automatic classification of toxic comments. We highlight that monolingual approaches for this task still outperform multilingual experiments and that large-scale datasets are needed for building reliable models. Also, we show that there are still challenges to be overcome, such as the naturally significant class imbalance when dealing with multi-label classification.

As future work, in addition to deal with class imbalance, we intend to evaluate if aggregating classes with high divergences between annotators can build more reliable models. Besides, we intend to assess the benefits of adding unlabelled data to ToLD-Br to use semi-supervised techniques.

## 7 Acknowledgements

We thank the volunteers from UFSCar that made this research possible. The MIDAS group<sup>12</sup> from the Federal University of São Carlos (UFSCar), Brazil, funded the annotation process. The SoBig-Data TransNational Access program (EU H2020, grant agreement: 654024) funded Diego Silva and João Leite's visits to the University of Sheffield.

<sup>12</sup>[midas.ufscar.br](http://midas.ufscar.br)

## References

- Ron Artstein and Massimo Poesio. 2008. Survey article: Inter-coder agreement for computational linguistics. *Computational Linguistics*, 34(4):555–596.
- Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, and Manuela Sanguinetti. 2019. SemEval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 54–63, Minneapolis, Minnesota, USA. Association for Computational Linguistics.
- Miguel Ángel Álvarez Carmona, Estefanía Guzmán-Falcón, Manuel Montes y Gómez, Hugo Jair Escalante, Luis Villaseñor Pineda, Verónica Reyes-Meza, and Antonio Rico Sulayes. 2018. Overview of mex-a3t at ibereval 2018: Authorship and aggressiveness analysis in mexican spanish tweets. In *Proceedings of the Third Workshop on Evaluation of Human Language Technologies for Iberian Languages (IberEval 2018)*, volume 2150 of *CEUR Workshop Proceedings*, pages 74–96. CEUR-WS.org.
- Yi-Ling Chung, Elizaveta Kuzmenko, Serra Sinem Tekiroglu, and Marco Guerini. 2019. CONAN - COunter NArratives through nichesourcing: a multilingual dataset of responses to fight online hate speech. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2819–2829, Florence, Italy. Association for Computational Linguistics.
- Thomas Davidson, Dana Warmsley, Michael W. Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. In *Proceedings of the 11th International AAAI Conference on Web and Social Media*, pages 512–515. AAAI Press.
- Rogers Prates de Pelle and Viviane P. Moreira. 2017. Offensive comments in the Brazilian web: a dataset and baseline results. In *Proceedings of the VI Brazilian Workshop on Social Network Analysis and Mining*, pages 510–519, Porto Alegre, RS, Brazil. SBC.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Tobias Springenberg, Manuel Blum, and Frank Hutter. 2019. Auto-sklearn: efficient and robust automated machine learning. In *Automated Machine Learning*, pages 113–134. Springer, Cham.
- Paula Fortuna, João Rocha da Silva, Juan Soler-Company, Leo Wanner, and Sérgio Nunes. 2019. A hierarchically-labeled Portuguese hate speech dataset. In *Proceedings of the Third Workshop on Abusive Language Online*, pages 94–104, Florence, Italy. Association for Computational Linguistics.
- Antigoni Maria Founta, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. 2018. Large scale crowdsourcing and characterization of twitter abusive behavior. In *Proceedings of the Twelfth International AAAI Conference on Web and Social Media*, pages 491–500, Stanford, California. AAAI Press.
- Thomas Mandl, Sandip Modha, Prasenjit Majumder, Daksh Patel, Mohana Dave, Chintak Mandlia, and Aditya Patel. 2019. Overview of the HASOC Track at FIRE 2019: Hate Speech and Offensive Content Identification in Indo-European Languages. In *Proceedings of the 11th Forum for Information Retrieval Evaluation*, page 14–17, Kolkata, India.
- Puneet Mathur, Rajiv Shah, Ramit Sawhney, and Debanjan Mahata. 2018. Detecting offensive tweets in Hindi-English code-switched language. In *Proceedings of the Sixth International Workshop on Natural Language Processing for Social Media*, pages 18–26, Melbourne, Australia. Association for Computational Linguistics.
- Hamdy Mubarak, Kareem Darwish, and Walid Magdy. 2017. Abusive language detection on Arabic social media. In *Proceedings of the First Workshop on Abusive Language Online*, pages 52–56, Vancouver, BC, Canada. Association for Computational Linguistics.
- Hala Mulki, Hatem Haddad, Chedi Bechikh Ali, and Halima Alshabani. 2019. L-HSAB: A Levantine twitter dataset for hate speech and abusive language. In *Proceedings of the Third Workshop on Abusive Language Online*, pages 111–118, Florence, Italy. Association for Computational Linguistics.
- Fabio Souza, Rodrigo Nogueira, and Roberto Lotufo. 2019. Portuguese named entity recognition using bert-crfs. *arXiv preprint arXiv:1909.10649*, pages 1–8.
- Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. 2009. Mining multi-label data. In *Data mining and knowledge discovery handbook*, pages 667–685. Springer.
- Zeerak Waseem and Dirk Hovy. 2016. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the NAACL Student Research Workshop*, pages 88–93, San Diego, California. Association for Computational Linguistics.
- Michael Wiegand, Melanie Siegel, and Josef Ruppenhofer. 2018. Overview of the GermEval 2018

**Shared Task on the Identification of Offensive Language.** In *Proceedings of GermEval 2018, 14th Conference on Natural Language Processing (KONVENS 2018)*, pages 1–10, Vienna, Austria.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierrette Cistac, Tim Rault, R’emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. **HuggingFace’s Transformers: State-of-the-art natural language processing.** *arXiv preprint arXiv:1910.03771*, abs/1910.03771:1–11.

Ellery Wulczyn, Nithum Thain, and Lucas Dixon. 2017. **Ex Machina: Personal Attacks Seen at Scale.** In *Proceedings of the 26th International Conference on World Wide Web*, page 1391–1399, Perth, Australia.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019a. **Predicting the type and target of offensive posts in social media.** In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1415–1420, Minneapolis, Minnesota. Association for Computational Linguistics.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019b. **SemEval-2019 task 6: Identifying and categorizing offensive language in social media (OffensEval).** In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 75–86, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2020. SemEval-2020 Task 12: Multilingual Offensive Language Identificationin Social Media (OffensEval 2020). In *To appear in the Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain.

# Measuring What Counts: The Case of Rumour Stance Classification

Carolina Scarton

University of Sheffield, UK

c.scarton@sheffield.ac.uk

Diego Furtado Silva

Federal University of São Carlos, Brazil

diegofs@ufscar.br

Kalina Bontcheva

University of Sheffield, UK

k.bontcheva.sheffield.ac.uk

## Abstract

Stance classification can be a powerful tool for understanding whether and which users believe in online rumours. The task aims to automatically predict the stance of replies towards a given rumour, namely *support*, *deny*, *question*, or *comment*. Numerous methods have been proposed and their performance compared in the RumourEval shared tasks in 2017 and 2019. Results demonstrated that this is a challenging problem since naturally occurring rumour stance data is highly imbalanced. This paper specifically questions the evaluation metrics used in these shared tasks. We re-evaluate the systems submitted to the two RumourEval tasks and show that the two widely adopted metrics – accuracy and macro- $F1$  – are not robust for the four-class imbalanced task of rumour stance classification, as they wrongly favour systems with highly skewed accuracy towards the majority class. To overcome this problem, we propose new evaluation metrics for rumour stance detection. These are not only robust to imbalanced data but also score higher systems that are capable of recognising the two most informative minority classes (*support* and *deny*).

## 1 Introduction

The automatic analysis of online rumours has emerged as an important and challenging Natural Language Processing (NLP) task. Rumours in social media can be defined as claims that cannot be verified as true or false at the time of posting (Zubiaga et al., 2018). Prior research (Mendoza et al., 2010; Kumar and Carley, 2019) has shown that the stances of user replies are often a useful predictor of a rumour’s likely veracity, specially in the case of false rumours that tend to receive a higher number of replies denying them (Zubiaga et al., 2016). However, their automatic classification is far from trivial as demonstrated by the

results of two shared tasks – RumourEval 2017 and 2019 (Derczynski et al., 2017; Gorrell et al., 2019). More specifically, sub-task A models rumour stance classification (RSC) as a four-class problem, where replies can:

- **support**/agree with the rumour;
- **deny** the veracity of the rumour;
- **query**/ask for additional evidence;
- **comment** without clear contribution to assessing the veracity of the rumour.

Figure 1 shows an example of a reply *denying* a post on Twitter.



Figure 1: Example of a *deny* stance.

In RumourEval 2017 the training data contains 297 rumourous threads about eight events. The test set has 28 threads, with 20 threads about the same events as the training data and eight threads about unseen events. In 2019, the 2017 training data is augmented with 40 Reddit threads. The new 2019 test set has 56 threads about natural disasters from Twitter and a set of Reddit data (25 threads). These datasets for RSC are highly imbalanced: the *comment* class is considerably larger than the other classes. Table 1 shows the distribution of stances per class in both 2017 and 2019 datasets, where 66% and 72% of the data (respectively) correspond to *comments*. *Comments* arguably are the

	<b>2017</b>	<b>2019</b>
support	1,004 (18%)	1,184 (14%)
deny	415 (7%)	606 (7%)
query	464 (8%)	608 (7%)
comment	3,685 (66%)	6,176 (72%)
total	5,568	8,574

Table 1: Distribution of stances per class – with percentages between parenthesis.

least useful when it comes to assessing overall rumour veracity, unlike *support* and *deny* which have been shown to help with rumour verification (Mendoza et al., 2010). Therefore, RSC is not only an imbalanced, multi-class problem, but it also has classes with different importance. This is different from standard stance classification tasks (e.g. SemEval 2016 task 6 (Mohammad et al., 2016)), where classes have arguably the same importance. It also differs from the veracity task (RumourEval sub-task B), where the problem is binary and it is not as an imbalanced problem as RSC.<sup>1</sup>

RumourEval 2017 evaluated systems based on accuracy (*ACC*), which is not sufficiently robust on imbalanced datasets (Huang and Ling, 2005). This prompted the adoption of macro-*F1* in the 2019 evaluation. Kumar and Carley (2019) also argue that macro-*F1* is a more reliable evaluation metric for RSC. Previous work on RSC also adopted these metrics (Li et al., 2019b; Kochkina et al., 2018; Dungs et al., 2018).

This paper re-evaluates the sub-task A results of RumourEval 2017 and 2019.<sup>2</sup> It analyses the performance of the participating systems according to different evaluation metrics and shows that even macro-*F1*, that is robust for evaluating binary classification on imbalanced datasets, fails to reliably evaluate the performance on RSC. This is particularly critical in RumourEval where not only is data imbalanced, but also two minority classes (*deny* and *support*) are the most important to classify well. Based on prior research on imbalanced datasets in areas other than NLP (e.g. Yijing et al. (2016) and Elrahman and Abraham (2013)), we propose four alternative metrics for evaluating RSC. These metrics change the systems ranking for RSC in RumourEval 2017 and 2019, rewarding systems with high performance on the minority classes.

<sup>1</sup>Other NLP tasks, like sentiment analysis are also not comparable, since these tasks are either binary classification (which is then solved by using macro-*F1*) or do not have a clear priority over classes.

<sup>2</sup>We thank the organisers for making the data available.

## 2 Evaluation metrics for classification

We define  $TP$  = true positives,  $TN$  = true negatives,  $FP$  = false positives and  $FN$  = false negatives, where  $TP_c$  ( $FP_c$ ) is equivalent to the true (false) positives and  $TN_c$  ( $FN_c$ ) is equivalent to the true (false) negatives for a given class  $c$ .

**Accuracy (*ACC*)** is the ratio between the number of correct predictions and the total number of predictions ( $N$ ):  $ACC = \frac{\sum_{c=1}^C TP_c}{N}$ , where  $C$  is the number of classes. *ACC* only considers the values that were classified correctly, disregarding the mistakes. This is inadequate for imbalanced problems like RSC where, as shown in Table 1, most of the data is classified as *comments*. As shown in Section 3, most systems will fail to classify the *deny* class and still achieve high scores in terms of *ACC*. In fact, the best system for 2017 according to *ACC* (Turing) fails to classify all *denies*.

**Precision ( $P_c$ ) and Recall ( $R_c$ )**  $P_c$  is the ratio between the number of correctly predicted instances and all the predicted values for  $c$ :  $P_c = \frac{TP_c}{TP_c + FP_c}$ .  $R_c$  is the ratio between correctly predicted instances and the number of instances that actually belongs to the class  $c$ :  $R_c = \frac{TP_c}{TP_c + FN_c}$ .

**macro- $F\beta_c$**   $F\beta_c$  score is defined as the harmonic mean of precision and recall, where the per-class score can be defined as:  $F\beta_c = (1 + \beta^2) \frac{P_c \cdot R_c}{\beta^2 P_c + R_c}$ . If  $\beta = 1$ ,  $F\beta$  is the *F1* score. If  $\beta > 1$ ,  $P$  is given a higher weight and if  $\beta < 1$ ,  $R$  is given a higher weight. The macro- $F\beta$  is the arithmetic mean between the  $F\beta$  scores for each class: macro- $F\beta_c = \frac{\sum_{c=1}^C F\beta_c}{C}$ . Although macro-*F1* is expected to perform better than *ACC* for imbalanced binary problems, its benefits in the scenario of multi-class classification are not clear. Specifically, as it relies on the arithmetic mean over the classes, it may hide the poor performance of a model in one of the classes if it performs well on the majority class (i.e. *comments* in this case). For instance, as shown in Table 2, according to macro-*F1* the best performing system would be *ECNU*, which still fails to classify correctly almost all *deny* instances.

**Geometric mean** Metrics like the geometric mean of  $R$ :

$$GMR = \sqrt[C]{\prod_{c=1}^C R_c}$$

	<i>ACC</i>	macro- <i>F1</i>	<i>GMR</i>	<i>wAUC</i>	<i>wF1</i>	<i>wF2</i>
Turing a	<b>0.784</b> (1)	0.434 (5)	0.000 (8)	0.583 (7)	0.274 (6)	0.230 (7)
UWaterloo (Bahuleyan and Vechtomova, 2017)	0.780 (2)	0.455 (2)	0.237 (5)	0.595 (5)	0.300 (2)	0.255 (6)
ECNU (Wang et al., 2017)	0.778 (3)	<b>0.467</b> (1)	0.214 (7)	0.599 (4)	0.289 (4)	0.263 (4)
Mama Edha (García Lozano et al., 2017)	0.749 (4)	0.453 (3)	0.220 (6)	<b>0.607</b> (1)	0.299 (3)	0.283 (3)
NileTMRG (Enayet and El-Beltagy, 2017)	0.709 (5)	0.452 (4)	<b>0.363</b> (1)	0.606 (2)	<b>0.306</b> (1)	<b>0.296</b> (1)
IKM (Chen et al., 2017)	0.701 (6)	0.408 (7)	0.272 (4)	0.570 (8)	0.241 (7)	0.226 (8)
IITP (Singh et al., 2017)	0.641 (7)	0.403 (8)	0.345 (2)	0.602 (3)	0.276 (5)	0.294 (2)
DFKI DKT (Srivastava et al., 2017)	0.635 (8)	0.409 (6)	0.316 (3)	0.589 (6)	0.234 (8)	0.256 (5)
majority class	0.742	0.213	0.000	0.500	0.043	0.047
all denies	0.068	0.032	0.000	0.500	0.051	0.107
all support	0.090	0.041	0.000	0.500	0.066	0.132

Table 2: Evaluation of RumourEval 2017 submissions. Values between parenthesis are the ranking of the system according to the metric. The official evaluation metric column (*ACC*) is highlighted in bold.

	<i>ACC</i>	<b>macro-<i>F1</i></b>	<i>GMR</i>	<i>wAUC</i>	<i>wF1</i>	<i>wF2</i>
BLCU NLP (Yang et al., 2019)	0.841 (2)	<b>0.619</b> (1)	0.571 (2)	0.722 (2)	<b>0.520</b> (1)	0.500 (2)
BUT-FIT (Fajcik et al., 2019)	<b>0.852</b> (1)	0.607 (2)	0.519 (3)	0.689 (3)	0.492 (3)	0.441 (3)
eventAI (Li et al., 2019a)	0.735 (11)	0.578 (3)	<b>0.726</b> (1)	<b>0.807</b> (1)	0.502 (2)	<b>0.602</b> (1)
UPV (Ghanem et al., 2019)	0.832 (4)	0.490 (4)	0.333 (5)	0.614 (5)	0.340 (4)	0.292 (5)
GWU (Hamidian and Diab, 2019)	0.797 (9)	0.435 (5)	0.000 (7)	0.604 (6)	0.284 (5)	0.265 (6)
SINAI-DL (García-Cumbreras et al., 2019)	0.830 (5)	0.430 (6)	0.000 (8)	0.577 (7)	0.255 (7)	0.215 (7)
wshuyi	0.538 (13)	0.370 (7)	0.467 (4)	0.627 (4)	0.261 (6)	0.325 (4)
Columbia (Liu et al., 2019)	0.789 (10)	0.363 (8)	0.000 (9)	0.562 (10)	0.221 (10)	0.191 (9)
jurebb	0.806 (8)	0.354 (9)	0.122 (6)	0.567 (9)	0.229 (8)	0.120 (12)
mukundyr	0.837 (3)	0.340 (10)	0.000 (10)	0.570 (8)	0.224 (9)	0.198 (8)
nxl	0.828 (7)	0.327 (11)	0.000 (11)	0.557 (11)	0.206 (11)	0.173 (10)
WeST (Baris et al., 2019)	0.829 (6)	0.321 (12)	0.000 (12)	0.551 (12)	0.197 (12)	0.161 (11)
Xinthl	0.725 (12)	0.230 (13)	0.000 (13)	0.493 (13)	0.072 (13)	0.071 (13)
majority class	0.808	0.223	0.000	0.500	0.045	0.048
all denies	0.055	0.026	0.000	0.500	0.042	0.091
all support	0.086	0.040	0.000	0.500	0.063	0.128

Table 3: Evaluation of RumourEval 2019 submissions. Values between parenthesis are the ranking of the system according to the metric. The official evaluation metric column (macro-*F1*) is highlighted in bold.

are proposed for evaluating specific types of errors. As *FNs* may be more relevant than *FPs* for imbalanced data, assessing models using *R* is an option to measure this specific type of error. Moreover, applying *GMR* for each class severely penalises a model that achieves a low score for a given class.

**Area under the *ROC* curve** Receiver operating characteristic (*ROC*) (Fawcett, 2006) assesses the performance of classifiers considering the relation between  $R_c$  and the false positive rate, defined as (per class):  $FPR_c = \frac{FP_c}{TN_c + FP_c}$ . Since RSC consists of discrete classifications, *ROC* charts for each  $c$  contain only one point regarding the coordinate ( $FPR_c, R_c$ ). Area under the *ROC* curve (*AUC*) measures the area of the curve produced by the points in an *ROC* space. In the discrete case, it measures the area of the polygon drawn by the segments connecting the vertices  $((0, 0), (FPR_c, R_c), (1, 1), (0, 1))$ . High *AUC* scores are achieved when *R* (probability of detection) is maximised, while *FPR* (probability of false alarm) is minimised. We experiment with a weighted variation of *AUC*:

$$wAUC = \sum_{c=1}^C w_c \cdot AUC_c.$$

**Weighted macro- $F\beta$**  a variation of macro- $F\beta$ , where each class also receives different weights, is also considered:

$$wF\beta = \sum_{c=1}^C w_c \cdot F\beta_c,$$

We use  $\beta = 1$  (*P* and *R* have the same importance) and  $\beta = 2$  (*R* is more important). Arguably, misclassifying *denies* and *supports* ( $FN_D$  and  $FN_S$ , respectively) is equivalent to ignore relevant information for debunking a rumour. Since *FNs* negatively impact *R*, we hypothesise that  $\beta = 2$  is more robust for the RSC case.

*wAUC* and *wF $\beta$*  are inspired by empirical evidence that different classes have different importance for RSC.<sup>3</sup> Weights should be manually defined, since they cannot be automatically learnt.

<sup>3</sup>Similarly, previous work proposes metrics (Elkan, 2001) and learning algorithms (Chawla et al., 2008) based on class-specific mis-classification costs.

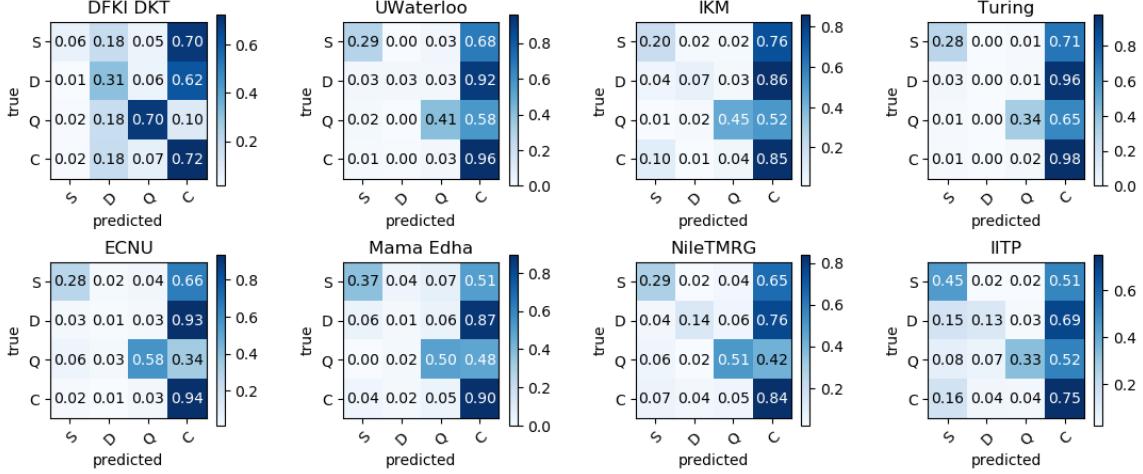


Figure 2: Confusion matrix for systems from RumourEval 2017.

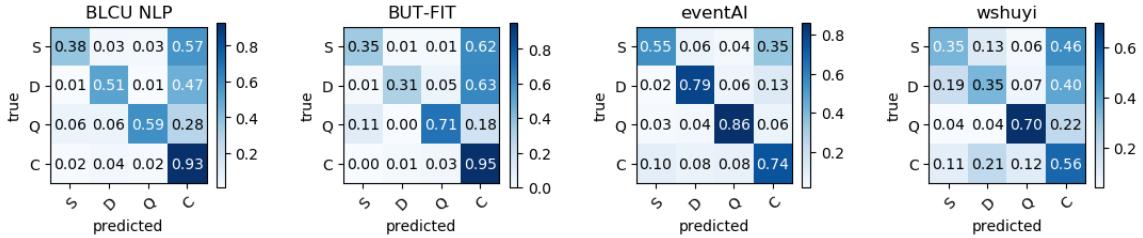


Figure 3: Confusion matrix for selected systems from RumourEval 2019. All other systems failed to classify correctly either all or the vast majority of *deny* instances.

We follow the hypothesis that *support* and *deny* classes are more informative than others.<sup>4</sup>

### 3 Re-evaluating RumourEval task A

Tables 2 and 3 report the different evaluation scores per metric for each of the RumourEval 2017 and 2019 systems.<sup>5</sup> *ACC* and macro-*F1* are reported in the second and third columns respectively, followed by a column for each of the four proposed metrics. Besides evaluating the participating systems, we also computed scores for three baselines: majority class (all stances are considered *comments*), all *denies* and all *support* (all replies are classed as *deny/support*).

Our results show that the choice of evaluation metric has a significant impact on system ranking. In RumourEval 2017, the winning system based on *ACC* was Turing. However, Figure 2 shows that this system classified all *denies* in

<sup>4</sup> $w_{\text{support}} = w_{\text{deny}} = 0.40$ ,  $w_{\text{query}} = 0.15$  and  $w_{\text{comment}} = 0.05$ .

<sup>5</sup>The systems HLT(HITSZ), LECS, magc, UI-AI, shaheyu and NimbusTwoThousand are omitted because they do not provide the same number of inputs as the test set.

correctly, favouring the majority class (*comment*). When looking at the macro-*F1* score, Turing is classified as fifth, whilst the winner is ECNU, followed by UWaterloo. Both systems also perform very poorly on *denies*, classifying only 1% and 3% of them correctly. On the other hand, the four proposed metrics penalise these systems for these errors and rank higher those that perform better on classes other than the majority one. For example, the winner according to *GMR*, *wF1* and *wF2* is NileTMRG that, according to Figure 2, shows higher accuracy on the *deny*, *support* and *query* classes, without considerably degraded performance on the majority class. *wAUC* still favours the Mama Edha system which has very limited performance on the important *deny* class. As is evident from Figure 2, NileTMRG is arguably the best system in predicting all classes: it has the highest accuracy for *denies*, and a sufficiently high accuracy for *support*, *queries* and *comments*. Using the same criteria, the second best system should be IITP. The only two metrics that reflect this ranking are *GMR* and *wF2*. In the case of *wF1*, the second system is UWaterloo,

which has a very low accuracy on the *deny* class.

For RumourEval 2019, the best system according to macro- $F1$  (the official metric) is BLCU NLP, followed by BUT-FIT. However, after analysing the confusion matrices in Figure 3, we can conclude that eventAI is a more suitable model due to its high accuracy on *support* and *deny*. Metrics  $GMR$ ,  $wAUC$  and  $wF2$  show eventAI as the best system. Finally, wshuyi is ranked as fourth according to  $GMR$ ,  $wAUC$  and  $wF2$ , while it ranked seventh in terms of macro- $F1$ , behind systems like GWU and SINAI-DL that fail to classify all *deny* instances. Although wshuyi is clearly worse than eventAI, BLCU NLP and BUT-FIT, it is arguably more reliable than systems that misclassify the large majority of *denies*.<sup>6</sup> Our analyses suggest that  $GMR$  and  $wF2$  are the most reliable for evaluating RSC tasks.

#### 4 Weight selection

In Section 3,  $wAUC$ ,  $wF1$  and  $wF2$  have been obtained using empirically defined weights ( $w_{support} = w_{deny} = 0.40$ ,  $w_{query} = 0.15$  and  $w_{comment} = 0.05$ ). These values reflect the key importance of the *support* and *deny* classes. Although *query* is less important than the first two, it is nevertheless more informative than *comment*.

Previous work tried to adjust the learning weights in order to minimise the effect of the imbalanced data. García Lozano et al. (2017) (Mama Edha), change the weights of their Convolutional Neural Network (CNN) architecture, giving higher importance to *support*, *deny* and *query* classes, to better reflect their class distribution.<sup>7</sup> Ghanem et al. (2019) (UPV) also change the weights in their Logistic Regression model in accordance with the data distribution criterion.<sup>8</sup> Nevertheless, these systems misclassify almost all *deny* instances.

Table 4 shows the RumourEval 2017 systems ranked according to  $wF2$  using the Mama Edha and UPV weights. In these cases,  $wF2$  benefits DFKI DKT, ranking it first, since *queries* receive a higher weight than *support*. However, this system only correctly classifies 6% of *support* instances, which makes it less suitable for our task than NileTMRG for instance. ECNU is also ranked

<sup>6</sup>Confusion matrices for all systems of RumourEval 2019 are presented in Appendix A.

<sup>7</sup> $w_{support} = 0.157$ ,  $w_{deny} = 0.396$ ,  $w_{query} = 0.399$  and  $w_{comment} = 0.048$

<sup>8</sup> $w_{support} = 0.2$ ,  $w_{deny} = 0.35$ ,  $w_{query} = 0.35$  and  $w_{comment} = 0.1$

better than Mama Edha and IITP, likely due to its higher performance on *query* instances.

	$wF2$ Mama Edha	$wF2$ UPV
Turing	0.246 (8)	0.289 (8)
UWaterloo	0.283 (7)	0.322 (5)
ECNU	0.334 (3)	0.364 (3)
Mama Edha	0.312 (4)	0.349 (4)
NileTMRG	0.350 (2)	0.374 (2)
IKM	0.293 (5)	0.318 (7)
IITP	0.289 (6)	0.321 (6)
DFKI DKT	<b>0.399 (1)</b>	<b>0.398 (1)</b>

Table 4: RumourEval 2017 evaluated using  $wF2$  with weights from Mama Edha and UPV.

Arguably, defining weights based purely on data distribution is not sufficient for RSC. Thus our empirically defined weights seem to be more suitable than those derived from data distribution alone, as the former accurately reflect that *support* and *deny* are the most important, albeit minority distributed classes. Further research is required in order to identify the most suitable weights for this task.

#### 5 Discussion

This paper re-evaluated the systems that participated in the two editions of RumourEval task A (stance classification). We showed that the choice of evaluation metric for assessing the task has a significant impact on system rankings. The metrics proposed here are better suited to evaluating tasks with imbalanced data, since they do not favour the majority class. We also suggest variations of  $AUC$  and macro- $F\beta$  that give different weights for each class, which is desirable for scenarios where some classes are more important than others.

The main lesson from this paper is that evaluation is an important aspect of NLP tasks and it needs to be done accordingly, after a careful consideration of the problem and the data available. In particular, we recommend that future work on RSC uses  $GMR$  and/or  $wF\beta$  (preferably  $\beta = 2$ ) as evaluation metrics. Best practices on evaluation rely on several metrics that can assess different aspects of quality. Therefore, relying on several metrics is likely the best approach for RSC evaluation.

#### Acknowledgments

This work was funded by the WeVerify project (EU H2020, grant agreement: 825297). The SoBigData TransNational Access program (EU H2020, grant agreement: 654024) funded Diego Silva’s visit to the University of Sheffield.

## References

- Hareesh Bahuleyan and Olga Vechtomova. 2017. **UWaterloo at SemEval-2017 task 8: Detecting stance towards rumours with topic independent features.** In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 461–464, Vancouver, Canada. Association for Computational Linguistics.
- Ipek Baris, Lukas Schmelzeisen, and Steffen Staab. 2019. **CLEARumor at SemEval-2019 task 7: ConvоЛving ELMo against rumors.** In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1105–1109, Minneapolis, Minnesota, USA. Association for Computational Linguistics.
- Nitesh V. Chawla, David A. Cieslak, Lawrence O. Hall, and Ajay Joshi. 2008. **Automatically countering imbalance and its empirical relationship to cost.** *Data Mining and Knowledge Discovery*, 17(2):225–252.
- Yi-Chin Chen, Zhao-Yang Liu, and Hung-Yu Kao. 2017. **IKM at SemEval-2017 task 8: Convolutional neural networks for stance detection and rumor verification.** In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 465–469, Vancouver, Canada. Association for Computational Linguistics.
- Leon Derczynski, Kalina Bontcheva, Maria Liakata, Rob Procter, Geraldine Wong Sak Hoi, and Arkaitz Zubiaga. 2017. **SemEval-2017 task 8: RumourEval: Determining rumour veracity and support for rumours.** In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 69–76, Vancouver, Canada. Association for Computational Linguistics.
- Sebastian Dungs, Ahmet Aker, Norbert Fuhr, and Kalina Bontcheva. 2018. **Can rumour stance alone predict veracity?** In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3360–3370, Santa Fe, New Mexico, USA. Association for Computational Linguistics.
- Charles Elkan. 2001. **The foundations of cost-sensitive learning.** In *Proceedings of the 17th International Joint Conference on Artificial Intelligence - Volume 2*, pages 973–978, Seattle, Washington, USA. International Joint Conferences on Artificial Intelligence.
- Shaza M. Abd Elrahman and Ajith Abraham. 2013. **A Review of Class Imbalance Problem.** *Journal of Network and Innovative Computing*, 1:332–340.
- Omar Enayet and Samhaa R. El-Beltagy. 2017. **NileTMRG at SemEval-2017 task 8: Determining rumour and veracity support for rumours on twitter.** In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 470–474, Vancouver, Canada. Association for Computational Linguistics.
- Martin Fajcik, Pavel Smrz, and Lukas Burget. 2019. **BUT-FIT at SemEval-2019 task 7: Determining the rumour stance with pre-trained deep bidirectional transformers.** In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1097–1104, Minneapolis, Minnesota, USA. Association for Computational Linguistics.
- Tom Fawcett. 2006. **An introduction to ROC analysis.** *Pattern Recognition Letters*, 27(8):861–874.
- Miguel A. García-Cumbreras, Salud María Jiménez-Zafra, Arturo Montejo-Ráez, Manuel Carlos Díaz-Galiano, and Estela Saquete. 2019. **SINAI-DL at SemEval-2019 task 7: Data augmentation and temporal expressions.** In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1120–1124, Minneapolis, Minnesota, USA. Association for Computational Linguistics.
- Marianela García Lozano, Hanna Lilja, Edward Tjörnhammar, and Maja Karasalo. 2017. **Mama edha at SemEval-2017 task 8: Stance classification with CNN and rules.** In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 481–485, Vancouver, Canada. Association for Computational Linguistics.
- Bilal Ghanem, Alessandra Teresa Cignarella, Cristina Bosco, Paolo Rosso, and Francisco Manuel Rangel Pardo. 2019. **UPV-28-UNITO at SemEval-2019 task 7: Exploiting post's nesting and syntax information for rumor stance classification.** In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1125–1131, Minneapolis, Minnesota, USA. Association for Computational Linguistics.
- Genevieve Gorrell, Elena Kochkina, Maria Liakata, Ahmet Aker, Arkaitz Zubiaga, Kalina Bontcheva, and Leon Derczynski. 2019. **SemEval-2019 task 7: RumourEval, determining rumour veracity and support for rumours.** In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 845–854, Minneapolis, Minnesota, USA. Association for Computational Linguistics.
- Sardar Hamidian and Mona Diab. 2019. **GWU NLP at SemEval-2019 task 7: Hybrid pipeline for rumour veracity and stance classification on social media.** In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1115–1119, Minneapolis, Minnesota, USA. Association for Computational Linguistics.
- Jin Huang and Charles X Ling. 2005. **Using AUC and accuracy in evaluating learning algorithms.** *IEEE Transactions on Knowledge and Data Engineering*, 17(3):299–310.
- Elena Kochkina, Maria Liakata, and Arkaitz Zubiaga. 2018. **All-in-one: Multi-task learning for rumour verification.** In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3402–3413, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

- Sumeet Kumar and Kathleen Carley. 2019. **Tree LSTMs with convolution units to predict stance and rumor veracity in social media conversations.** In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5047–5058, Florence, Italy. Association for Computational Linguistics.
- Quanzhi Li, Qiong Zhang, and Luo Si. 2019a. **even-tAI at SemEval-2019 task 7: Rumor detection on social media by exploiting content, user credibility and propagation information.** In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 855–859, Minneapolis, Minnesota, USA. Association for Computational Linguistics.
- Quanzhi Li, Qiong Zhang, and Luo Si. 2019b. **Rumor detection by exploiting user credibility information, attention and multi-task learning.** In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1173–1179, Florence, Italy. Association for Computational Linguistics.
- Zhuoran Liu, Shivali Goel, Mukund Yelahanka Raghuprasad, and Smaranda Muresan. 2019. **Columbia at SemEval-2019 task 7: Multi-task learning for stance classification and rumour verification.** In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1110–1114, Minneapolis, Minnesota, USA. Association for Computational Linguistics.
- Marcelo Mendoza, Barbara Poblete, and Carlos Castillo. 2010. **Twitter under crisis: can we trust what we RT?** In *Proceedings of the First Workshop on Social Media Analytics*, pages 71–79, Washington, DC, USA. Association for Computing Machinery.
- Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. 2016. **SemEval-2016 task 6: Detecting stance in tweets.** In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 31–41, San Diego, California. Association for Computational Linguistics.
- Vikram Singh, Sunny Narayan, Md Shad Akhtar, Asif Ekbal, and Pushpak Bhattacharyya. 2017. **IITP at SemEval-2017 task 8 : A supervised approach for rumour evaluation.** In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 497–501, Vancouver, Canada. Association for Computational Linguistics.
- Ankit Srivastava, Georg Rehm, and Julian Moreno Schneider. 2017. **DFKI-DKT at SemEval-2017 task 8: Rumour detection and classification using cascading heuristics.** In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 486–490, Vancouver, Canada. Association for Computational Linguistics.
- Feixiang Wang, Man Lan, and Yuanbin Wu. 2017. **ECNU at SemEval-2017 task 8: Rumour evalua-**
- tion using effective features and supervised ensemble models.** In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 491–496, Vancouver, Canada. Association for Computational Linguistics.
- Ruoyao Yang, Wanying Xie, Chunhua Liu, and Dong Yu. 2019. **BLCU\_NLP at SemEval-2019 task 7: An inference chain-based GPT model for rumour evaluation.** In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1090–1096, Minneapolis, Minnesota, USA. Association for Computational Linguistics.
- Li Yijing, Guo Haixiang, Liu Xiao, Li Yanan, and Li Jinling. 2016. **Adapted ensemble classification algorithm based on multiple classifier system and feature selection for classifying multi-class imbalanced data.** *Knowledge-Based Systems*, 94:88–104.
- Arkaitz Zubiaga, Ahmet Aker, Kalina Bontcheva, Maria Liakata, and Rob Procter. 2018. **Detection and resolution of rumours in social media: A survey.** *ACM Computing Surveys*, 51(2):32:1–32:36.
- Arkaitz Zubiaga, Maria Liakata, Rob Procter, Geraldine Wong Sak Hoi, and Peter Tolmie. 2016. **Analysing how people orient to and spread rumours in social media by looking at conversational threads.** *Plos One*, 11(3).

## A Confusion matrices for all RumourEval 2019 systems

For completeness, Figure 4 shows the confusion matrices of all systems submitted to RumourEval 2019. Apart from the four systems discussed in Section 3, all other systems fails to correctly classify the large majority of *deny* instances.

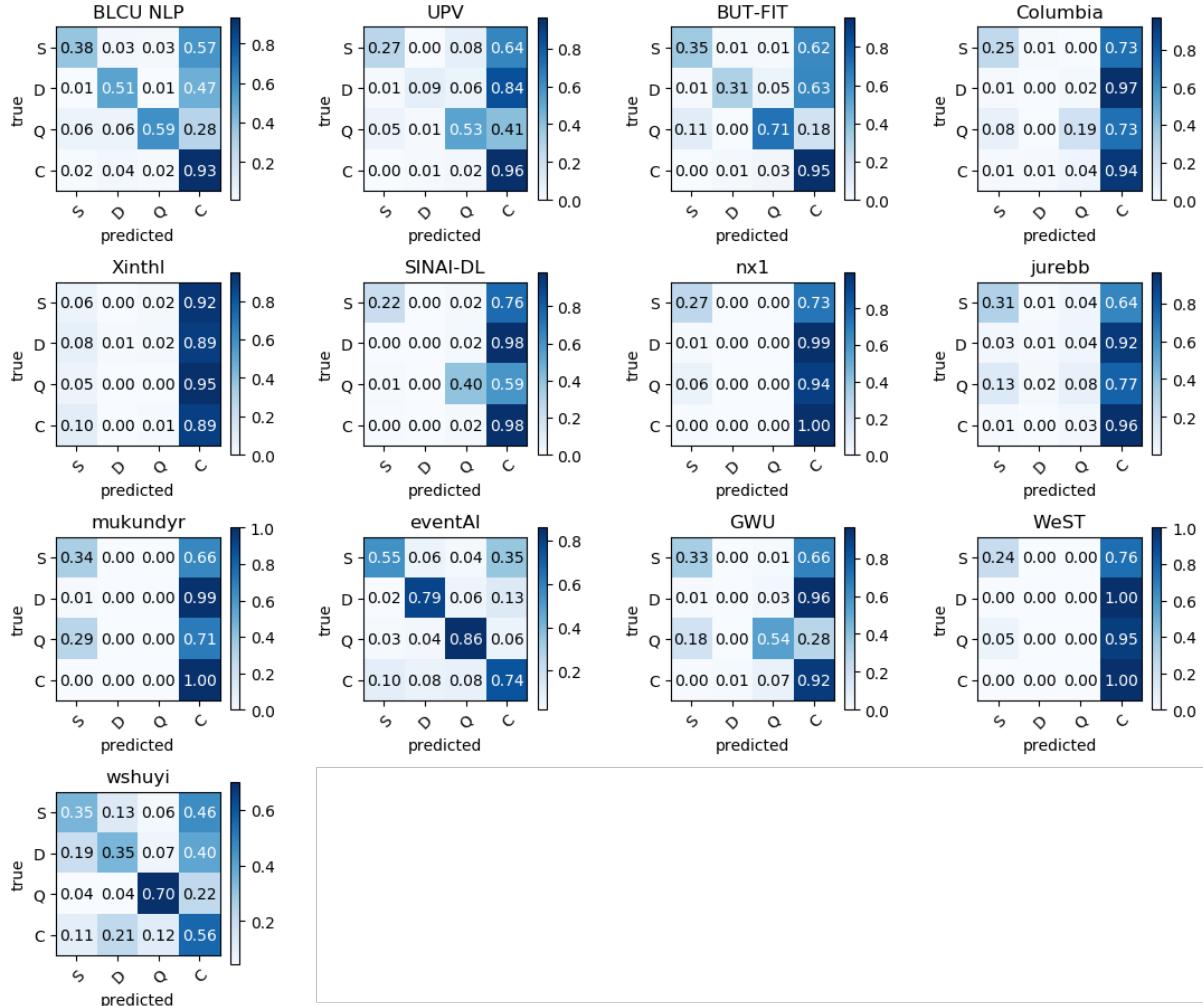


Figure 4: Confusion matrix for all systems from RumourEval 2019.

# Author Index

- Aharonov, Ranit, 447  
Ahmad, Zishan, 303  
Aletras, Nikolaos, 804  
Amplayo, Reinald Kim, 409  
Awadallah, Ahmed Hassan, 551
- Bahar, Syafri, 843  
Baldwin, Timothy, 598  
Banea, Carmen, 425  
Bateni, Peyman, 664  
Ben Noach, Matan, 884  
Bennett, Paul, 551  
Benz, Yannik, 786  
Bhattacharyya, Pushpak, 281, 303, 858, 900  
Biester, Laura, 425  
Bing, Lidong, 258  
Blache, Philippe, 224  
Blain, Frédéric, 366  
Bollegala, Danushka, 873  
Bontcheva, Kalina, 914, 925  
Bowman, Samuel R., 557, 672  
Brahman, Faeze, 588
- Cahyawijaya, Samuel, 843  
Calixto, Iacer, 504, 557  
Callison-Burch, Chris, 328  
Cardie, Claire, 551  
Carenini, Giuseppe, 516, 626, 664  
Chang, Kai-Wei, 609  
Chang, Walter, 529  
Chaturvedi, Snigdha, 588  
Chaudhary, Vishrav, 366  
Chauhan, Dushyant Singh, 281  
Chen, Haotian, 637  
Chen, Pei, 811  
Chen, Ruijie, 672  
Chen, Xiao, 191  
Chen, Yubo, 181, 811  
Chen, Yufeng, 726  
Chen, Yun, 191  
Chersoni, Emmanuele, 224  
Chi, Zewen, 12  
Cho, Kyunghyun, 334  
Choi, Jinho D., 358
- Chu, Eric, 643  
Chua, Tat-Seng, 122  
Cohen, Shay B., 378  
Cui, Xia, 873
- Dai, Wenliang, 269  
Danilevsky, Marina, 447  
Deleu, Johannes, 821  
Demeester, Thomas, 821  
Dernoncourt, Franck, 529  
Develder, Chris, 821  
Ding, Yuning, 347  
Dong, Li, 12, 87  
Du, Xinya, 551
- Eger, Steffen, 786  
Ekbal, Asif, 281, 303, 900  
El-Kishky, Ahmed, 366, 616
- Fei, Hao, 100  
Feng, Qihang, 70  
Feng, Yukun, 80  
Feng, Zhifan, 735  
FitzGerald, Jack, 576  
Fomicheva, Marina, 366  
Fourney, Adam, 551  
Fu, Zihao, 258  
Fung, Pascale, 269, 843
- Gao, Tianyu, 745  
Gao, Tong, 491  
Gao, Yingbo, 212, 389  
Garg, Siddhant, 460  
Ge, Tao, 201  
Glass, James, 334  
Goldberg, Yoav, 884  
Goldstein, Felicia, 358  
Gong, Ming, 687  
Gosangi, Rakesh, 706  
Guo, Mengfei, 726  
Guo, Yingmei, 37  
Gupta, Deepak, 900  
Gupta, Vivek, 706  
Guz, Grigorii, 664  
Guzmán, Francisco, 366, 616

- Hackinen, Brad, 626  
Hajjar, Ihab, 358  
Han, Xu, 169, 745  
Hao, Yaru, 87  
Hayashibe, Yuta, 890  
He, Tianxing, 334  
He, Yuan, 378  
Hernandez Abrego, Gustavo, 435  
Herold, Christian, 212  
Horbach, Andrea, 347  
Hou, Lei, 770  
Htut, Phu Mon, 557  
Hu, Chenlong, 80  
Huang, Chu-Ren, 224, 833  
Huang, Gabriel, 470  
Huang, Guoping, 1  
Huang, Heyan, 12  
Huang, Junhong, 70  
Huang, Kuan-Hao, 609  
Huang, Minlie, 122, 248, 770  
Huang, Qi, 491  
Huang, Ruihong, 811  
Huang, Shaohan, 248  
Huang, Xuedong, 536  
Huang, Yan, 122  
Huang, Yongfeng, 44, 181  
Huang, Zhong-Yu, 720  
  
Iwakura, Tomoya, 154  
  
Jang, Seongbo, 133  
Ji, Donghong, 100  
Ji, Haozhe, 248  
Jia, Shengyu, 169  
Jiang, Wenbin, 726, 735  
Jiang, Xin, 191  
Jiang, Yiwei, 821  
Jin, Lifeng, 396  
Joshi, Manish, 781  
Jung, Dawoon, 133  
  
Kamigaito, Hidetaka, 80  
Kann, Katharina, 557  
Kano, Ryuji, 291  
Kanojia, Diptesh, 858  
Kanouchi, Shin, 890  
Kao, Hung-Yu, 18, 143  
Katsis, Yannis, 447  
Katsumata, Satoru, 163, 827  
Kawas, Ban, 447  
Ke, Pei, 248  
Keller, Frank, 409  
  
Kim, Doo Soon, 529  
Kim, Taeuk, 409  
Kim, Yu-Seop, 63  
Koehn, Philipp, 582  
Komachi, Mamoru, 163, 827  
Koto, Fajri, 598  
Kumar, Vishwajeet, 781  
Kurosawa, Michiki, 163  
  
Lam, Wai, 258, 542, 696  
Lau, Jey Han, 598  
Le, Yuquan, 54  
Lebanoff, Logan, 529  
Lee, Cheng-Syuan, 720  
Lee, Joohong, 133  
Leite, João Augusto, 914  
Lenci, Alessandro, 224  
Lenka, Pabitra, 900  
Li, Bowen, 409  
Li, Chen, 609  
Li, Chenliang, 201  
Li, Jiawen, 18  
Li, Juanzi, 169, 770  
Li, Liangyou, 191  
Li, Meng Yuan, 70  
Li, Peng, 169, 745  
Li, Renxuan Albert, 358  
Li, Tianrui, 687  
Li, Weikang, 106  
Li, Xi, 637  
Li, Xiaohong, 843  
Li, Ying, 726  
Li, Yuan-Fang, 781  
Liang, Bowen, 435  
Liang, Yingyu, 460  
Liao, Keng-Te, 720  
Liao, Lizi, 122  
Lim, Zhi Yuan, 843  
Lin, Shou-de, 720  
Lin, Yankai, 745  
Liu, Fei, 529  
Liu, Haokun, 557  
Liu, Kang, 811  
Liu, Lemao, 1  
Liu, Qun, 191  
Liu, Zhiyuan, 169, 745, 770  
Liu, Zihan, 269  
Liu, Zitao, 122  
Lorré, Jean-Pierre, 313  
Lu, Jing, 653  
Lu, Qin, 833  
Luo, Gan, 770

- Luo, Huaishao, 687  
Lyu, Qing, 328  
Lyu, Yajuan, 726, 735  
  
Ma, Xutai, 582  
Mahata, Debanjan, 706  
Mahendra, Rahmad, 843  
Mao, Xian-Ling, 12  
Mathias, Sandeep, 858  
Matsushita, Kyoumoto, 154  
Mihalcea, Rada, 425  
Milewski, Victor, 504  
Mishra, Abhijit, 858  
Miura, Yasuhide, 291  
Moens, Marie-Francine, 504  
Mooney, Raymond, 491  
Muglich, Darius, 664  
Murthy, Rudra, 858  
  
Nadeem, Moin, 334  
Narayanan Sundararaman, Mukuntha, 303  
Neishi, Masato, 890  
Ney, Hermann, 212, 389  
Ng, Vincent, 653  
Ninomiya, Takashi, 154  
  
Oh, Byoung-Doo, 63  
Ohkuma, Tomoko, 291  
Okazaki, Naoaki, 890  
Okumura, Manabu, 80  
Okura, Shumpei, 116  
Omote, Yutaro, 154  
Ono, Shingo, 116  
Opitz, Juri, 235  
Ouchi, Hiroki, 890  
  
Pang, Bo, 470  
Parekh, Zarana, 435  
Park, Kyubyong, 133  
Peng, Hao, 745  
Petrusca, Alexandru, 588  
Phang, Jason, 557  
Pino, Juan, 582  
Preotiuc-Pietro, Daniel, 804  
Pruksachatkun, Yada, 557  
Pugoy, Reinald Adrian, 143  
Purwarianti, Ayu, 843  
  
Qi, Tao, 44, 181  
Qian, Kun, 447  
  
Ramakrishnan, Ganesh, 781  
Rambelli, Giulia, 224  
  
Ren, Yafeng, 100  
Renduchintala, Adithya, 366  
Rivera, Clara, 470  
Roy, Deb, 643  
  
S R, Dhanush, 281  
Sánchez Villegas, Danae, 804  
Sasaki, Mei, 116  
Scarton, Carolina, 914, 925  
Schuler, William, 396  
Sen, Prithviraj, 447  
Shah, Rajiv Ratn, 706  
Shang, Guokan, 313  
Sharma, Rohit Kumar, 460  
Shi, Bei, 258  
Shi, Yu, 536, 687  
Shou, Linjun, 687  
Silva, Diego, 914, 925  
Sim, Robert, 551  
Sneyd, Alison, 759  
Soleman, Sidik, 843  
Soriciut, Radu, 470  
Specia, Lucia, 366  
Stent, Amanda, 706  
Stevenson, Mark, 759  
Su, Qi, 833  
Sujana, Yudianto, 18  
Sun, Maosong, 745  
Sun, Shuo, 366  
Sung, Yunhsuan, 435  
Swaminathan, Avinash, 706  
  
Takamura, Hiroya, 80  
Tamura, Akihiro, 154  
Tang, Jie, 770  
Taniguchi, Tomoki, 291  
Tixier, Antoine, 313  
Trebbi, Francesco, 626  
Tu, Kewei, 93  
  
Uppal, Shagun, 706  
  
Vania, Clara, 557, 672  
Vazirgiannis, Michalis, 313  
Vijayaraghavan, Prashanth, 643  
Vincentio, Karissa, 843  
  
Wadhwa, Sahil, 637  
Wan, Mingyu, 833  
Wang, Baoxun, 70  
Wang, Chenyu, 770  
Wang, Hongfei, 163  
Wang, Jin, 27

- Wang, Qi, 735  
Wang, Qian, 1  
Wang, Taifeng, 811  
Wang, Wei, 435  
Wang, Weiyue, 212, 389  
Wang, Xiaozhi, 169  
Wang, Xinyu, 93  
Wang, Zongsheng, 70  
Wei, Furu, 12, 87, 201, 248  
Wenjie, Ying, 54  
Wilie, Bryan, 843  
Winata, Genta, 843  
Wu, Bowen, 70  
Wu, Chuhan, 44, 181  
Wu, Fangzhao, 44  
Wu, Yunfang, 106  
Wu, Zhiyong, 37
- Xiang, Rong, 833  
Xiao, Chaojun, 745  
Xiao, Wen, 516  
Xing, Linzi, 626  
Xiong, Hantao, 54  
Xu, Canwen, 201  
Xu, Jinan, 726  
Xu, Ke, 87  
Xu, Mingxing, 37  
Xu, Ruochen, 536
- Yang, Hang, 811  
Yang, Yaoliang, 745  
Yang, Yinfai, 435  
Yang, Zijian, 212, 389  
Yu, Jifan, 770  
Yu, Liang-Chih, 27  
Yu, Qian, 696  
Yu, Tiezheng, 269  
Yuan, Li, 27  
Yuan, Yifei, 542  
Yuan, Zhigang, 181
- Zaporojets, Klim, 821  
Zeng, Michael, 536  
Zesch, Torsten, 347  
Zhang, Haimin, 706  
Zhang, Haiyang, 759  
Zhang, Huan, 70  
Zhang, Jiajun, 1  
Zhang, Li, 328  
Zhang, Wexuan, 696  
Zhang, Xuejie, 27  
Zhang, Zheng, 122
- Zhao, Jun, 811  
Zhou, Jie, 169, 745  
Zhou, Jingbo, 542  
Zhu, Chenguang, 536  
Zhu, Xiaoyan, 122  
Zhu, Yong, 726, 735  
Zhu, Zhenhai, 470  
Zong, Chengqing, 1  
Zukov Gregoric, Andrej, 637